QUATERNION-BASED DUAL LOOP NONLINEAR TRAJECTORY CONTROL OF QUADROTORS

A Thesis
Presented to
The Academic Faculty

By

Joo-Won Kang

In Partial Fulfillment
of the Requirements for the Degree
Master of Science in the
School of Mechanical Engineering

Georgia Institute of Technology

May 2021

QUATERNION-BASED DUAL LOOP NONLINEAR TRAJECTORY CONTROL OF QUADROTORS

Approved by:

Dr. Nader Sadegh, Advisor
School of Mechanical Engineering
Georgia Institute of Technology

Dr. Aldo A. Ferri
School of Mechanical Engineering
Georgia Institute of Technology

Dr. Jonathan D. Rogers
School of Aerospace Engineering/Mechanical Engineering
Georgia Institute of Technology

Date Approved: April 23, 2021

ACKNOWLEDGMENTS

Much of the work completed here would not have been possible without the help of the people named here. I would first like to thank my thesis advisor Dr. Nader Sadegh for giving me full support and helping me understand different control theories throughout the preparation of this work. The work is much his as as it is mine as he helped me expand many of the unproven implementation methods we came up with so that the methods were theoretically sound. Dr. Sadegh's great attention to detail and perfectionism allowed the final work to be theoretically well-grounded which naturally led to positive experimental results. I would also like to thank Chase Urschel who helped me to understand the material during the initial stages of this work. Much of the foundation of the experimental setup used in preparation of this work was largely due to him.

I would also like to thank Dr. Aldo A. Ferri and Dr. William Singhose for both providing interesting research topics and financially funding me that both widened my understanding of different control theories and systems and, at the same time, allowed me to work on this study without having any financial difficulties. I would also like to acknowledge Dr. Ferri and Dr. Jonathan D. Rogers for participating as committee members for this thesis and spending time to review and evaluate the presented work.

Finally, I would like to thank my family and friends for all the support they have gave me to overcome difficulties I faced during my time here and throughout life.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF ACRONYMS

DCM Direction Cosine Matrices

EKF Extended Kalman Filter

PD Proportional-Derivative

PI Proportional-Integral

PID Proportional-Integral-Derivative

PWM Pulse-width Modulation

RC Radio-Controlled

SITL Software in the Loop

UAVs Unmanned Aerial Vehicles

VTOL Vertical Take-Off and Landing

SUMMARY

Many multirotor controllers achieve globally stable attitude control through the usage of quaternions, as it does not inherent any singularities, unlike other alternatives such as Euler angles. However, globally stable position or velocity control of quadrotors have rarely been achieved; most controllers limit their attitude within a certain range to avoid singularities that occur during position control of quadrotors. This thesis focuses on presenting a globally stable quaternion-based dual loop nonlinear control scheme. Globality of the control scheme is both achieved in 1) the sense that all errors exponentially converge to zero regardless of the initial state errors and 2) the controller can stabilize at and pass through any point of attitude during position control.

The two control loops are structured so that the outer loop controls the translational velocity or position, and the inner loop controls the attitude. The outer control loop uses a Proportional-Integral (PI) feedback structure. The proportional action is in terms of the translational position or velocity of the quadrotor, and the integral action is primarily used to eliminate steady-state error of the control state. The inner attitude control loop is a Proportional-Derivative (PD) feedback loop, where the proportional action is in terms of the vector component of the quaternion representation of the quadrotor attitude and the derivative action is in terms of the quadrotor angular velocity. Additional control gains, other than the PID gains, are employed to achieve globality in the Lyapunov sense. Furthermore, nonlinear feedforward terms, which acts as compensations for the nonlinear dynamics of the quadrotor and gyroscopic torques, were also employed to guarantee the global stability of the quadrotor.

This thesis also investigates different singularity and ambiguity issues, more specifically singularities that can arise during attitude control while trying to achieve global stability; these singularities, when not dealt with, may cause the controller to lose control of the quadrotor leading to instability at different locations. The thesis presents solutions to

these singularities which allows the quadrotor to achieve global position control so that the quadrotor can achieve any attitude without losing stability.

The results section shows that once all solutions are employed robust and accurate control of the quadrotor can be achieved. The thesis presents the effectiveness of the solutions to design a globally position tracking quadrotor controller with both simulation and implementation results. In particular, the quadrotor is seen tracking an ambitious linear position-based trajectory which commands the quadrotor to complete 360° flipping motions in various directions. The robustness of the control scheme is shown with the fact that all trajectories were completed without the need for changing control parameters for each individual trajectory.

CHAPTER 1
INTRODUCTION

Interests in multirotors have been growing in the research community since the late 2000s. Multirotors make excellent Unmanned Aerial Vehicles (UAVs) due to their capability of maintaining a fixed position and Vertical Take-Off and Landing (VTOL). Fixed-wing UAVs have longer flight times but need runways for take-off and landing and to move continuously to generate lift. Due to their unique characteristics, multirotor UAVs have become popular for military applications, photography and filming, and agricultural spraying and surveying.

1.1 Motivation and Objectives

Quadrotors are interesting and fairly complex systems with six degrees of freedom: three translational degrees of freedom and three rotational degrees of freedom. The fact that all six degrees of freedom must be manipulated with only four control forces, three torque forces and one thrust force, does not make the control of quadrotors any simpler. In recent years, the control of the general movement of quadrotors have developed from the control of attitude only, which only covers the three rotational degrees of freedom, while having the remaining three positional degrees of freedom to be controlled in an open-looped manner to complete control of all six degrees of freedom. Even with the recent developments, global control of quadrotors in the sense that it can attain any attitude from only translational trajectories have rarely been achieved. Due to the nature of the most popular rotational conventions, singularities are bound to occur. The stability of the control of quadrotors may deteriorate if such singularities are introduced in the process. Another difficulty is the fact that the three translational degrees of freedom are not directly coupled with all three rotational degrees of freedom. During translational control of quadrotors,

the direction of the thrust force is the only component of the quadrotor that directly effects the actual translational movement of the quadrotor, which can largely be determined by two of the rotational degrees of freedom that controls the tilt of the quadrotor. Designing a global controller while making sure the control laws stably control all six degrees of freedom is challenging as it requires the controller to incorporate the remaining single degree of freedom smoothly and without singularity. Many studies avoid such singularities by limiting the range of control; however, in order to achieve a truly global controller such singularities must be addressed.

In light of the considerations discussed above, the objectives of this study can be summarized as follows:

- Design a controller that is global in the sense that all state errors converge to zero no matter how large the initial error is.
- Design a controller that is global in the sense that all attitudes can be achieved.
- Investigate singularities that may occur during the process of achieving global control, and solve such issues in a smooth manner.
- Incorporate all considerations and ensure that the overall controller is fast and robust.

1.2 Current Control Strategies

Attitude control of rigid bodies, including quadrotors, commonly use one of the three rotational conventions: Euler angles, Direction Cosine Matrices (DCM), and quaternions. Each attitude has their own advantages and disadvantages which makes them suitable to use for different purposes. The Euler angle convention consists of three angular values each representing the elementary rotation around the three principal axes: roll, pitch, and yaw. Some examples of quadrotor controllers which utilize the Euler angle convention can be found in [1], [2], and [3]. DCMs are singular rotation matrices which

combines three elemental rotation matrices around the three principal axes. [4] and [5] are two examples of quadrotor controllers which use DCMs as their main attitude convention. Quaternion attitude is another alternative which has unique advantages that makes it preferable over the other two conversions. First of all, Quaternions do not divide rotations into three separate rotations, as the Euler Angles do, which makes it singular free. Additionally, quaternions only have four elements which makes it less computationally expensive than the DCM convention which includes nine. As a result, many attitude controllers of the quadrotor utilize quaternions. In [6], the author demonstrates how quaternions can be used effectively to control rigid body attitude. [7] presents quaternion-based PD^2 with additional feedforward compensation attitude controller of a quadrotor. [8] introduces a non-linear P^2 controller was proposed. Finally, [9] presents a quaternion based PD attitude controller with compensations for uncertainties.

Although sole control of quadrotor attitude is much faster and more efficient, accurate control of all six degrees of freedom of the quadrotor is difficult to achieve. Without directly controlling the position of the quadrotor, dealing with disturbances, such as wind or other forces, may be challenging. More importantly, the ultimate purpose of controlling UAVs is to guarantee that the aircraft travels from one position to another without manual assistance, which is a feature that is difficult to achieve by controlling position indirectly through attitude controllers. The position control problem of quadrotors has been addressed by several papers using various control techniques, both linear and nonlinear. Proportional-Integral-Derivative (PID) control is one of the more commonly used techniques in quadrotors. [2] introduces a PID-based dual loop position tracking controller with angle saturation. [10] proposes a dual loop PD-PD structure control scheme to control the position of a quadrotor. Other than the commonly used PID control method, controllers based on control theories more suitable for nonlinear systems has also been explored and implemented. Sliding mode control can be found in numerous controllers. [1] introduces an adaptive sliding mode position controller, which is compared with a controller based on feedback

linearization; the adaptive law included in the control scheme allows the overall control to experience reduced chattering, which is a common disadvantage of sliding mode-based controllers. [11] presents a sliding mode controller based on Lyapunov functions which controls the altitude and attitude of the quadrotor. [12] proposes a sliding mode control which utilizes saturation functions instead of sign functions to reduce chattering. [13] introduces a dual-loop controller with a LQR based outer position control loop and an integral sliding mode attitude control to reduce the effects of uncertainties and disturbances. Finally, [14] presents a position tracking dual-loop controller which implements the sliding mode control method in both the inner attitude and outer position control loop. Back-stepping control is another control theory that has been studied by various papers. [15] introduces a seven step backstepping position controller of quadrotors guaranteed to achieve asymptotic stability based on the Lypaunov stability analysis. [16] introduces a Lyapunov based nine step backstepping attitude controller with angle limitations. Feedback linearization is another nonlinear control technique that have been explored in existing controllers. [17] introduces an asymptotically stable nonlinear feedback linearizing controller with slight backstepping. [18] introduces a Ricattie equation based LQR feedback linearized position controller of the quadrotor. [19] presents a dual loop position controller in which the outer loop is a proportional based controller with nonlinear compensation and the inner attitude loop is based on feedback linearization. The implementation of H_∞ control has also been explored in the past. [20] combines a nonlinear H_∞ based nonlinear controller with a backstepping translational trajectory controller. [21] applies nonlinear H_∞ control on the inner attitude loop with a trajectory tracking predictive controller in the outer translational loop. [22] presents a method of designing a H_∞ based linear controller with model parameters obtained from an in-depth system identification process. Furthermore, adaptive control has also been utilized to deal with modelling uncertainties and disturbances. [3] presents an adaptive control scheme to deal with modelling inertial uncertainties. [23] proposes an asymptotically stable control law, in which adaptive control is employed to eliminate

the need to collect angular velocity measurements during control. [24] presents a dual loop adaptive controller to compensate for dynamic changes of the center of gravity of the quadrotor. [25] presents a survey on different control methods and compares the advantages and disadvantages of different linear and nonlinear based techniques. Other than the more traditional linear and nonlinear control, several other control techniques have also been explored. [26] introduces a PI position control loop combined with a Self-Tuning Regulator to decrease response overshoot. More recently, [27] presented a controller that combined embedded model control with feedback linearization to control the quadrotor position. Neural network based has also been explored in [28] and [29], as well as controllers based on reinforcement learning as presented in [30].

Global quadrotor control has rarely been achieved. To track most position trajectories, a locally stable controller that is robust within the upper region of the 3D attitude space is sufficient; most simple trajectories do not require large changes in attitude, which, as a result, do not invoke many of the difficulties present in designing global controllers. There certainly are controllers tailored towards tracking aggressive trajectories which may have the capability of achieving global control. Although these controllers do indeed have the speed and accuracy to track agile maneuvers, many do not have clear solutions to singularities that occur during global tracking. [31] has actuator constraints that prevents the quadrotor from achieving certain attitude configurations as the controller focuses on tracking dynamically feasible trajectories. [32] utilizes sequences of trajectory tracking and attitude control to achieve aggressive flipping maneuvers. [33] focuses on rejecting aerodynamic disturbances without considering singularities that may occur while achieving global trajectory tracking control. [34] effectively deals with the singularity that occurs when tracking perching trajectories (i.e., achieving attitude configurations where the thrust direction is almost horizontal); however, other singularities that occur when the aircraft is inverted are not considered. Finally, [35] presents a dual loop control scheme: an outer attitude loop and an inner angular velocity tracking loop. Although the paper presents results

that demonstrate tracking of aggressive maneuvers, such as flipping motions, it focuses on attitude control, not position control, in which singularities primarily discussed throughout the thesis do not occur.

[5], [36], [37], and [38] are some of the few controllers that to a certain extent do have the versatility and capability to track more ambitious trajectories. [5] introduces globally stable backstepping controller which uses perturbation and smoothing functions to deal with undesirable singularities. However, globality cannot be achieved without additional restrictions as the overall closed-loop error system is shown to have two equilibrium points once the proposed control scheme is applied. More importantly, the paper does not include any simulation or experimental results to demonstrate such globality, which is only proven in theory. [36], on the other hand, proposes a DCM controller which is only almost global as it requires limitations on initial conditions. Also, the flipping motion, which can demonstrate the globality of the controller, were performed with sole attitude control suggesting the control scheme is not robust to execute such motions with position control of the quadrotor. Furthermore, [37] proposes a global converging controller that generates feasible trajectories based on alternating minimization; however, the letter does not consider the singularities that occur due to the differential flatness of the quadrotor and, at the same time, the presented results do not show any indication that the control method can achieve attitude configurations globally without running into those singularities. Finally, [38] presents a quadrotor that can perform upside-down perching. Although the presented controller does not demonstrate global state convergence, the upside-down perching maneuver does show that the controller effectively deals with all singularities that occurs while achieving such motion. Even so, the strategy involves stopping the motor when close to singular pose to prevent the introduction of singularity into the control inputs; this method limits the control of the quadrotor within such singular regions which may be undesirable for certain applications. The novelty of the controller presented in this study is in that many of these aggressive maneuvers can be achieved with appropriate designed trajectories as all

singularities that occur while tracking global trajectories are appropriately dealt with. Also, the strategy of avoiding singularity does not involve limiting control within singular regions which is another great advantage.

1.3 Thesis Formulation

[39] presents part of the work discussed during this thesis. Although sections of these publications will not be directly copied into the thesis, many of the concepts and proofs will be borrowed. This thesis investigates position or velocity control of a globally, exponentially stable quadrotor. Chapter 2 provides mathematical background used extensively throughout the thesis, which includes discussion on rotation conventions, singularities, basic Lyapunov stability, and quadrotor modelling. Chapter 3 presents the general control scheme of the quadrotor to achieve the objectives discussed above; in addition, the chapter investigates the different singularities that inevitably occurs due to the nature of the system. The chapter also includes an extensive Lyapunov stability analysis which shows the global exponential stability of the controller. Chapters 4 and 5 present the simulation and experimental performance of the controller implemented to a commercial quadrotor. The results of the efforts to achieve a global controller is discussed in this chapter as well. Finally, Chapter 6 presents a conclusion for the thesis.

CHAPTER 2
BACKGROUND

2.1 Different Angle Conventions

As discussed before, multirotor control is commonly based on Euler angles, direction cosine matrices (DCMs), or quaternions. In fact, many robotic research areas that deal with robot body orientation utilizes the three abovementioned conventions. Euler angles have the advantage of being the most intuitive, as each of the three Euler angles represent elementary rotations around the three principles axes of the aircraft: roll, pitch, and yaw. An alternative to the traditional Euler angles is the DCM representation. DCM representations are usually the combination of elementary rotation matrices that each represent the corresponding roll, pitch, and yaw. Though commonly used, Euler angles have inherent geometric singularities, commonly known as "Gimbal Lock", while DCM representations are computationally expensive as all nine elements of the matrix are utilized [8]. Quaternion attitude representation, on the other hand, presents a singularity-free and less computationally expensive alternative. All rotations essentially contain information about the axis of rotation and angle of rotation. In fact, the 3 conventions are merely mathematical conventions with different pros and cons to allow mathematical calculations of the rotations.

2.1.1 Euler Angles

Euler Angles are the combination of three elemental rotations around three principal axes of a fixed frame. The three rotations when combined can represent any attitude around the fixed frame and can reach any target attitude. The definition of Euler Angles may differ depending on the three principal axes chosen and the order of the three rotations. The axes selected can either be intrinsic or extrinsic: intrinsic axes are defined on the body-

fixed frame (i.e., the rotated frames after each elementary rotation), while extrinsic axes are always defined on the original fixed global frame.

Two types of rotation sequences can be used to achieve a target attitude. The first being the Proper, or Classic, Euler Angles which rotates around three axes, wherein the first and the third elemental rotations are around the same axis (i.e., Z-Y-Z or Z-X-Z). On the other hand, the Tait-Bryan convention, which is widely utilized in aerospace applications, rotates around three distinct elemental axes (i.e., Z-Y-X or Z-X-Y). Though different in method, the basic idea of the Euler Angles is that three rotations are sufficient to represent any rotation around the original fixed frame. As all fixed frames are consisted of 3 principal axes, a total of 12 different sequences can be used to represent attitude via the Euler Angle convention. However, due to the nature of the problem, as the thesis deals with aircrafts, the Tait-Bryan convention, specifically Euler Angles in the sequence of the extrinsic Z-Y-X, will be the center of discussion; whenever the thesis uses the term 'Euler Angles', it would be referring to the Tait-Bryan convention (Z-Y-X).

2.1.1.1 Tait-Bryan (Intrinsic Z-Y-X)

The Tait-Bryan convention is also known as the 'Roll-Pitch-Yaw' convention. The roll, pitch, and yaw angle each denote the rotation around the x,y and z axes, respectively. Figure 2.1 shows how Euler Angle sequences can be used to represent the orientation, body frame $\mathcal{A}$, of a quadrotor relative to a fixed frame $\mathcal{I}$. The frames $\mathcal{I}' = \{e'_x \;\; e'_y \;\; e'_z\}$ and $\mathcal{I}'' = \{e''_x \;\; e''_y \;\; e''_z\}$ are the intermediate frames after the first and second rotation occurs. The three Euler Angles (i.e., roll, pitch, and yaw) represent the rotation around the three principal axes, e''_x, e'_y , and e_z. Note that the order of rotation of the Intrinsic Z-Y-X convention is ZYX in the respective order; the first rotation is around the e_z axis with a rotation angle of γ (red), followed by a rotation around the e'_y axis with a rotation angle of β (blue), and finally a rotation around the e''_x axis with a rotation angle of α (green).

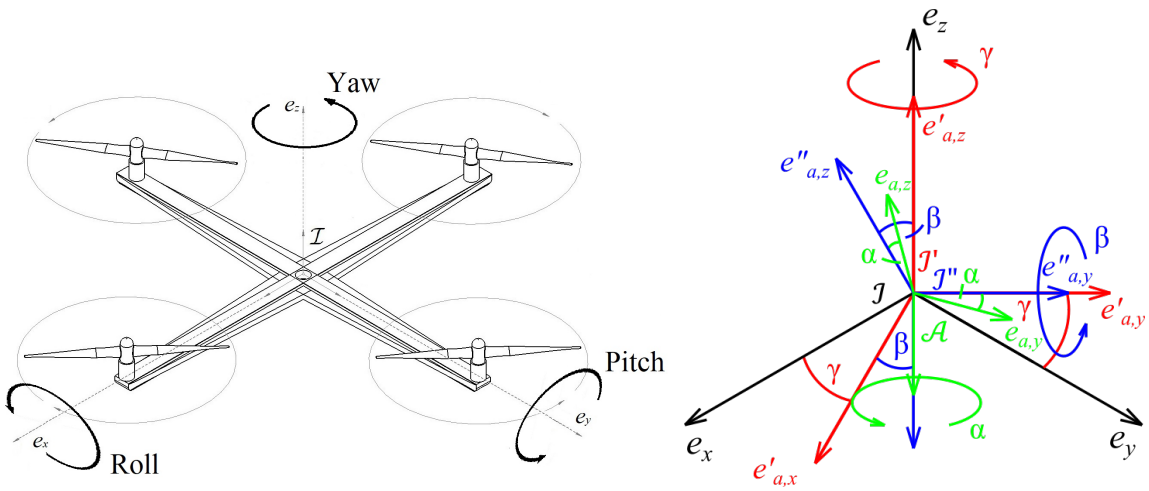


Figure 2.1: Euler Angles (Intrinsic Z-Y-X) Convention

2.1.1.2 Pros and Cons

Euler Angles, out of the three conventions, are the most intuitive as they essential represent the three rotation angles around three fixed axes. However, as intuitive as they are, there are many drawbacks to this convention. As mentioned before, all attitude conventions are merely a mathematical representation of the attitude which can be simplified into a rotation axis and a rotation angle. In the case of Euler Angles, the attitude is represented with three rotation axes and three rotation angles. This introduces complications as it is difficult to simplify the 3 consecutive rotations into a single axis/angle rotation by using Euler Angles alone; this can certainly be done but conversion into the other two conventions is required to do so. To put this into context, the method in which Euler Angles represent attitude, essentially the information of rotating the original frame $\mathcal{I}$, with the three principal axes, e_x, e_y, and e_z, to the final body-fixed frame $\mathcal{A}$, with the three principal axes, $e_{a,x}$, $e_{a,y}$, and $e_{a,z}$, can be shown with the Rodrigues' rotation formula:

$$v' = v\cos\hat{\theta} + (\hat{r}\times v)\sin\hat{\theta} + r\,(\hat{r}\cdot v)\left(1-\cos\hat{\theta}\right) \tag{2.1}$$

where v and v' each denote a 3×1 vector before and after rotation, and $\hat{r}$ and $\hat{\theta}$ each denote the axis of rotation and angle of rotation, respectively. Figure 2.2 visualizes a rotation

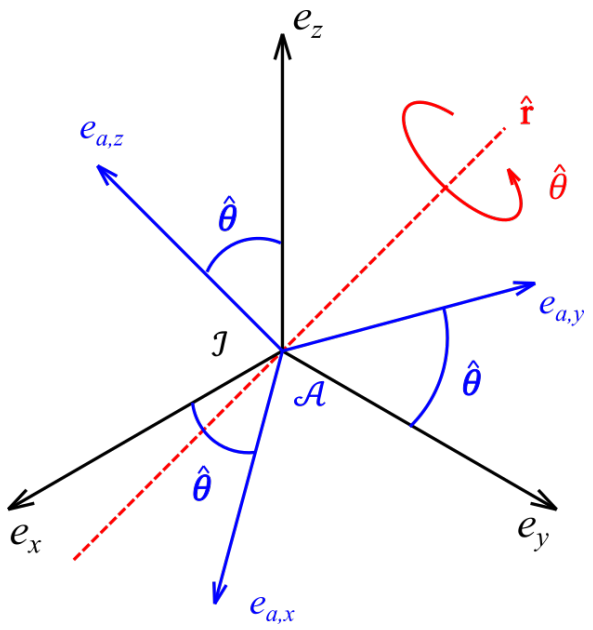


Figure 2.2: Rotation of $\hat{\theta}$ around $\hat{r}$

around an axis $\hat{r}$ with an angle of rotation of $\hat{\theta}$. Now let the three rotations axes each be $e_x = [1\ \ 0\ \ 0]^T$, $e_y = [0\ \ 1\ \ 0]^T$, and $e_z = [0\ \ 0\ \ 1]^T$, and the three Euler Angles each be α, β, and γ. With Equation 2.1, the full rotation can be expressed as the following:

$$
\begin{aligned}
e'_{a,x} &= e_x \cos\gamma + (e_z \times e_x)\sin\gamma + e_z\,(e_z \cdot e_x)\,(1-\cos\gamma) \\
e'_{a,y} &= e_y \cos\gamma + (e_z \times e_y)\sin\gamma + e_z\,(e_z \cdot e_y)\,(1-\cos\gamma) \\
e'_{a,z} &= e_z \cos\gamma + (e_z \times e_z)\sin\gamma + e_z\,(e_z \cdot e_z)\,(1-\cos\gamma)
\end{aligned}
$$

$$
\begin{aligned}
e''_{a,x} &= e'_{a,x} \cos\beta + \left(e'_{a,y} \times e'_{a,x}\right)\sin\beta + e'_{a,y}\left(e'_{a,y} \cdot e'_{a,x}\right)(1-\cos\beta) \\
e''_{a,y} &= e'_{a,y} \cos\beta + \left(e'_{a,y} \times e'_{a,y}\right)\sin\beta + e'_{a,y}\left(e'_{a,y} \cdot e'_{a,y}\right)(1-\cos\beta) \\
e''_{a,z} &= e'_{a,z} \cos\beta + \left(e'_{a,y} \times e'_{a,z}\right)\sin\beta + e'_{a,y}\left(e'_{a,y} \cdot e'_{a,z}\right)(1-\cos\beta)
\end{aligned}
\tag{2.2}
$$

$$
\begin{aligned}
e_{a,x} &= e''_{a,x} \cos\alpha + \left(e''_{a,x} \times e''_{a,x}\right)\sin\alpha + e''_{a,x}\left(e''_{a,x} \cdot e''_{a,x}\right)(1-\cos\alpha) \\
e_{a,y} &= e''_{a,y} \cos\alpha + \left(e''_{a,x} \times e''_{a,y}\right)\sin\alpha + e''_{a,x}\left(e''_{a,x} \cdot e''_{a,y}\right)(1-\cos\alpha) \\
e_{a,z} &= e''_{a,z} \cos\alpha + \left(e''_{a,x} \times e''_{a,z}\right)\sin\alpha + e''_{a,x}\left(e''_{a,x} \cdot e''_{a,z}\right)(1-\cos\alpha)
\end{aligned}
$$

As can be seen in Equation 2.2, the process of rotating a frame to another using Euler Angles alone is quite complicated; even so, the process can be simplified when used with other conventions. Another drawback that can be seen with the derivation are the unavoidable singularities that exist in Euler Angles, commonly known as "Gimbal Lock". The singularity occurs when the second angle of rotation, β in the case of the Intrinsic ZYX Euler Angle, is $\pm\frac{\pi}{2}$. The rotated frame at the singularity based on Equation 2.2, can be given as:

$$\begin{aligned} e_{a,x} &= \begin{bmatrix} 0 & \sin(\alpha+\gamma) & \cos(\alpha+\gamma) \end{bmatrix}^T \\ e_{a,y} &= \begin{bmatrix} 0 & \cos(\alpha+\gamma) & \sin(\alpha+\gamma) \end{bmatrix}^T \\ e_{a,z} &= \begin{bmatrix} 1 & 0 & 0 \end{bmatrix}^T \end{aligned} \tag{2.3}$$

Even though Euler Angles are well-defined for all orientations, countless combinations of α and γ, according to Equation 2.3, can lead to the same orientation; hence, the rotation at singularity is non-unique which makes it difficult to use. Consequently, many Euler Angle based controllers limit the attitude so that the singularity points do not cause instability during control [2, 16].

2.1.2 Direction Cosine Matrix (DCM)

Direction Cosine Matrices, or DCMs, are 3×3 orthogonal rotation matrices with determinants of 1 that represent the orientation of a body with respect to a reference frame. A DCM that represents a rotation the axis of rotation, $\hat{r}$, and angle of rotation, $\hat{\theta}$, can be expressed as follows [40]:

$$R = \cos\hat{\theta} I_{3\times 3} + \sin\hat{\theta} \left[\hat{r}\right]_\times + \left(1 - \cos\hat{\theta}\right)\left(\hat{r}\hat{r}^T\right) \tag{2.4}$$

where $[v]_\times$ represents a skew symmetric matrix equivalent to the cross-product operation of any vector $v \in \mathbb{R}^3$, which can be expressed as:

$$[v]_\times = \begin{bmatrix} 0 & -v_3 & v_2 \\ v_3 & 0 & -v_1 \\ -v_2 & v_1 & 0 \end{bmatrix} \tag{2.5}$$

Note that DCMs can also be viewed as the combination of the three body-fixed axes of the rotated orientation, $R = [e_{a,x} \;\; e_{a,y} \;\; e_{a,z}]$. This is important to note as the following sections of the thesis will be dealing with the quadrotor thrust vector, u, which directly aligned with the z-axis of the body fixed frame after rotation, $e_{a,z}$.

2.1.2.1 Pros and Cons

Although the DCM convention is not as intuitive as the Euler Angles, the convention has many advantages that makes it more favorable than the Euler Angles. Unlike the Euler Angles, the DCM convention does not divide a single rotation into three separate rotations. As a result, the DCM convention is free from ambiguity, in other words, all orientation only corresponds to one DCM. The DCM is also free from the singularity issue making it a good candidate when attempting to achieve a global controller. Nevertheless, a 3×3, which contains a total of 9 elements, is computationally expensive when compared with other conventions [8]. To add to this, additional steps are required to use DCMs to compute control inputs, as the final control torque inputs which all quadrotors receive only has 3 elements; steps are required to reduce the information contained in the 9 elements of the 3×3 matrix into 3 control torque inputs. [4] shows an example of the process of implementing DCM into quadrotor control in detail; using DCM in control, as can be seen in [4], adds complications that neither the Euler Angle nor the quaternion conventions require.

2.1.2.2 Mathematical Background

The transpose of a rotation matrix (i.e., R^T) is equivalent to the inverse of the same rotation matrix (i.e., R^{-1}) as all rotation matrices are orthogonal; hence, the transpose will be used instead of the inverse as it is much easier to compute.

The time derivative of any rotation matrix can also be simply obtained with the following equation:

$$\dot{R} = [\omega]_\times R \tag{2.6}$$

where $[\omega]_\times$ is a skew-symmetric matrix as defined in Equation 2.5.

2.1.3 Quaternion

Quaternions are a 4D number system consisted of real scalar parts (q_0) and imaginary vector parts (q_v). Quaternions are used for a variety of applications which include aerospace practice. The quaternion representation of aircraft attitude can be expressed as a 4-element vector, with which a rotation around a unit vector $\hat{r} \in \mathbb{R}^3$ with an angle of $\hat{\theta}$ can be represented as [7, 40]:

$$\begin{bmatrix} q_0 \\ q_v \end{bmatrix} = \begin{bmatrix} \cos\left(\frac{\hat{\theta}}{2}\right) \\ \hat{r}\sin\left(\frac{\hat{\theta}}{2}\right) \end{bmatrix}. \tag{2.7}$$

Quaternions, when used to represent orientation, by definition, as evident in Equation 2.7, are unit quaternions, which forces them to satisfy the following constraint:

$$q^T q = q_0^2 + q_v^T q_v = \cos^2\left(\frac{\hat{\theta}}{2}\right) + \hat{r}^2 \sin^2\left(\frac{\hat{\theta}}{2}\right) = 1. \tag{2.8}$$

2.1.3.1 Pros and Cons

The quaternion convention, unlike the Euler Angle convention, is singular free for most attitudes. Quaternions, when compared with DCMs, are also much simpler, and less computationally expensive, to implement in an attitude controller as it only contains

four elements. However, quaternion also has inherent drawbacks. The main drawback of quaternions is that they have a sign ambiguity; in other words, two quaternions with opposite signs can lead to the same rotation ([q_0,q_v] and [-q_0,-q_v]). The sign ambiguity of quaternions poses a problem when calculating quaternion errors as the rotation from one orientation to another would be possible with two quaternions of opposite signs. However, this drawback is often commonly avoided by simply constraining q_0 to only positive real numbers ($0 \leq q_0 \leq 1$). [41] also presents an non-ambiguous algorithm to convert rotation matrices to quaternions. Additionally, the quaternion representation of attitude requires a unity constraint, which may pose a problem from certain applications [40].

The thesis focuses on building a controller based on quaternion attitude as: 1) quaternions are non-singular, thus fits the objective the controller designed attempts to achieve, 2) it is much simpler to implement and computationally less expensive, than the DCM convention, and 3) the disadvantages of using this method are mostly avoidable with simple modifications.

2.1.3.2 Mathematical Background

The following section lists some important quaternion mathematical operations and definitions that may be pertinent to the remainder of the thesis. To avoid ambiguity, all quaternion attitudes used in this study will be constrained by the following condition:

$$0 \leq q \leq 1. \tag{2.9}$$

The inversion of quaternions can be performed as follows:

$$q^{-1} = \frac{q^*}{q^T q} \tag{2.10}$$

where $q^* = [q_0 \ \ \text{-}q_v]^T$ denotes the conjugate of the quaternion.

The time derivative of a unit quaternion q representing the attitude of a rigid body

is given by

$$\dot{q} = \frac{1}{2} q \bar{\omega}_b \tag{2.11}$$

where $\bar{\mathbf{v}} = [0 \ \mathbf{v}^T]^T$ denotes the quaternion equivalent of any vector $\mathbf{v} \in \mathbb{R}^3$ [40, 6]. Note that the angular velocity vector used is the body-fixed angular velocity, ω_b. A similar expression can be found in terms of the angular velocity relative to the global frame:

$$\dot{q} = \frac{1}{2} \bar{\omega} q$$

Additionally, throughout the thesis we shall denote the rotation matrix corresponding to the quaternion q by R_q:

$$\overline{R_q \mathbf{v}} = q \bar{\mathbf{v}} q^{-1}, \forall \mathbf{v} \in \mathbb{R}^3 \tag{2.12}$$

where $\bar{*}$ is a quaternion which consists of a zero scalar a part and $*$ as its vector part (i.e., $\bar{*} = [0 \ *^T]^T$).

A particular quaternion of interest is one that rotates unit vector $\mathbf{v} \in \mathbb{R}^3$ to another unit vector $\mathbf{w} \in \mathbb{R}^3$ free of any yaw (i.e., rotation about $\mathbf{v}$ or $\mathbf{w}$); such quaternions will be referred as *minimal* and denoted by $q_{\min}(\mathbf{v}, \mathbf{w})$. The axis of rotation of $q_{\min}(\mathbf{v}, \mathbf{w})$ is perpendicular to both vectors with the corresponding rotation angle of $\hat{\theta} = \cos^{-1}(\mathbf{v}^T \mathbf{w})$. If $\hat{\theta} \neq \pi$, or equivalently $\mathbf{v}^T \mathbf{w} \neq -1$, then it can be seen that

$$q_{\min}(\mathbf{v}, \mathbf{w}) = \frac{1}{\sqrt{2(1 + \mathbf{v}^T \mathbf{w})}} \begin{bmatrix} 1 + \mathbf{v}^T \mathbf{w} \\ \mathbf{v} \times \mathbf{w} \end{bmatrix} \tag{2.13}$$

is the unique minimal unit quaternion that rotates $\mathbf{v}$ to $\mathbf{w}$.

2.2 Singularities

The following section describe basic backgrounds on singularity, or discontinuity, of real functions. The theory of singularity may be expanded to complex functions; however, require more complicated analysis which is not essential to this thesis.

2.2.1 Definition

Singularities occurs at any point of a real function, or its derivatives, that loses continuity. However, any discontinuity that occurs in the derivative are singularities that belong to the derivative of the function, and not the original function. Hence, the thesis shall define any point of a real function that is discontinuous up to two derivatives to be singular and unfit to be used in a global controller.

2.2.2 Types of Discontinuity

Each of the singularities, or discontinuities, discusses in this thesis can be categorized into three different types of discontinuities [42]: removable discontinuity, jump discontinuity, and essential discontinuity. All singularities dealt with during the remainder of the thesis can be characterized to the three above-mentioned discontinuities.

2.2.2.1 Removable Discontinuity

If function $f(x)$ has a point c at which the limit exists but does not equal the actual value of $f(x)$ at point c, or $f(x)$ is undefined at c, then $f(x)$ would be considered to have a *removable* discontinuity at point c. As shown in Figure 2.3, $f(x)$ is undefined at c but the limit at c, $\lim_{x\to c} f(x)$ exists, which makes it a removable discontinuity. Removable discontinuities are relatively easier to deal with, when compared with other discontinuities, as the undefined or discontinuous point can simply be replaced with the limit.

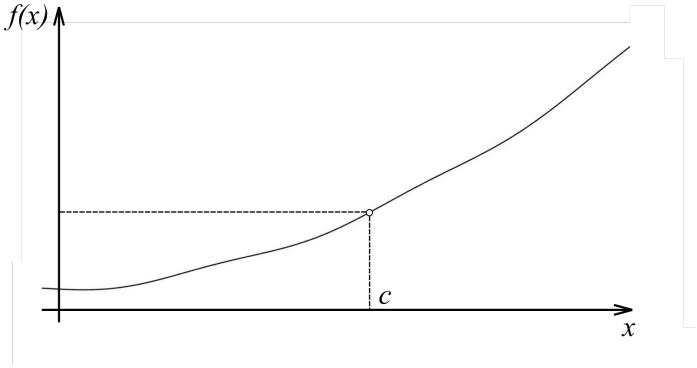
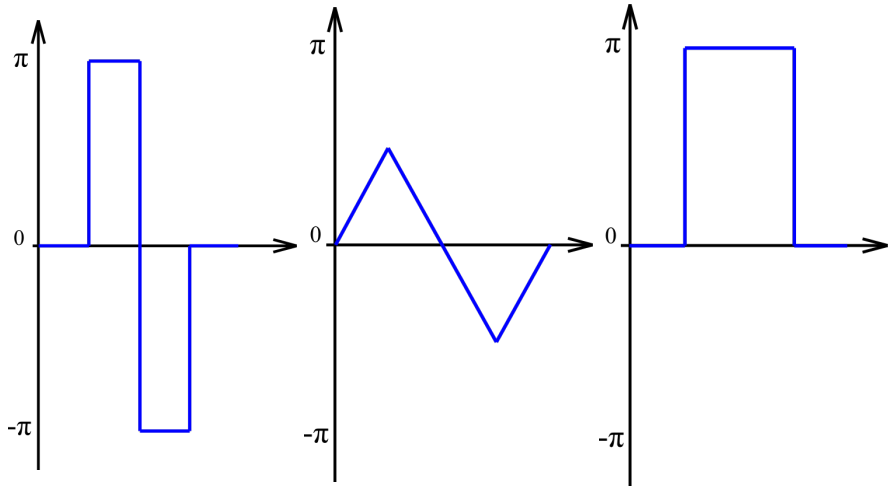


Figure 2.3: Example of Removable Discontinuity

2.2.2.2 Jump Discontinuity

If function $f(x)$ has a point c at which the left and right side limits exists but do not equal each other (i.e., , $\lim_{x\to c^-} f(x) \neq \lim_{x\to c^+} f(x)$), then $f(x)$ would be considered to have a *jump* discontinuity at point c. In fact, the Euler Angles are a good example of jump discontinuities. Figure 2.4 displays the three Euler angles as a body rotates around the y-axis with an angle of 2π. As the pitch, rotation around the y-axis, reaches $\pm\frac{\pi}{2}$, both the roll and yaw angles jump from 0 to π.

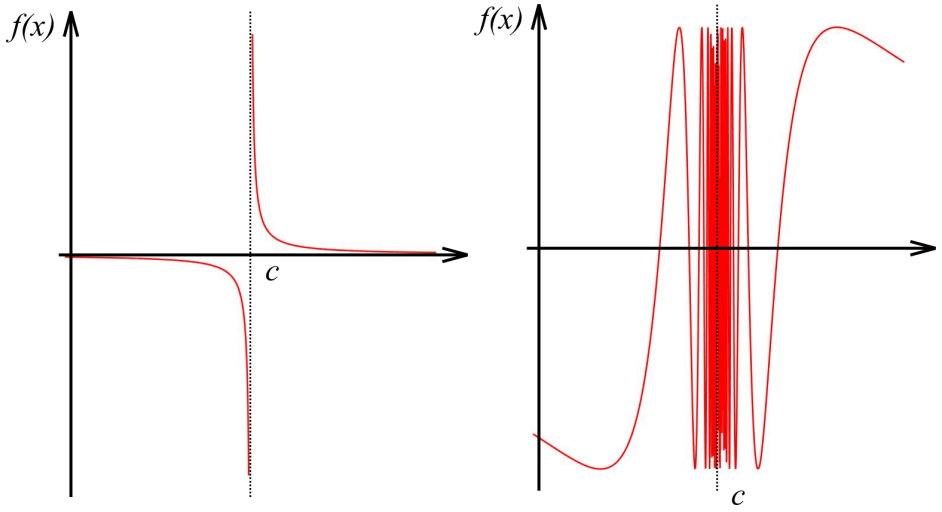


(a) Roll (b) Pitch (c) Yaw

Figure 2.4: Example of Jump Discontinuity

2.2.2.3 Essential Discontinuity

If function $f(x)$ has a point c at which the either the left or right side limit does not exist, then $f(x)$ would be considered to have an *essential* discontinuity at point c. The limit can cease to exist for various reasons; Figure 2.5 shows two examples of functions that have essential discontinuities at point c. Figure 2.5a is an example of infinite discontinuity; the left and right side limit at point c approaches infinity ($\pm\infty$). Figure 2.5b, on the other hand, is an example of oscillatory discontinuity; the limit at point c oscillates and does not converge to a single value.



(a) Plot of $f(x) = \frac{1}{(x+c)}$ (b) Plot of $f(x) = \sin\frac{1}{(x+c)}$

Figure 2.5: Example of Essential Discontinuity

2.3 Lyapunov Stability Theory

This section will briefly review definitions of the stability of equilibrium points. Although applicable to all systems, the definitions of Lyapunov stability reviewed in this section will be limited to autonomous systems as it is sufficient for the purpose of this study.

2.3.1 *Unstable to Global Exponential Stability*

An equilibrium point is considered stable if solutions starting near the equilibrium remain close to the equilibrium point, and unstable if the solutions diverge away from the equilibrium point. If the solution converges to the equilibrium point, then the equilibrium point is considered asymptotically stable. Finally, exponential stability, which is a subset of asymptotic stability, guarantees the solution converge to the equilibrium point at a much faster exponential decay rate [43, 44]. The definition of the four types of Lyapunov stability may be summarized as follows. Let $\dot{x} = f(x)$ be an autonomous nonlinear system with state $x(t)$, R and r each be the radius of two arbitrary regions where $\|x(t)\| < R, \forall t \geq 0$ and $\|x(t=0)\| < r$. For simplicity, assume the equilibrium state of the system $f(x)$ to be $x = 0$. Then,

- If such an r does not exist, the equilibrium state is said to be *unstable*.
- If such an r does exist, for any R, the equilibrium state is said to be *stable*.
- If such an r does exist and $\lim_{t\to\infty} x(t) = x(t=0)$, the equilibrium state is said to be *asymptotically stable*.
- If such an r does exist, $\lim_{t\to\infty} x(t) = x(t=0)$, and two constants $\zeta > 0$ and $\upsilon > 0$ exist such that $\|x(t)\| \leq \zeta\|x(t=0)\|e^{-\upsilon t}, \forall t \geq 0$, the equilibrium state is said to be *exponentially stable*.

Global stability, asymptotic or exponential, can be established if each stability is guaranteed for any initial state. If convergence is not guaranteed for certain initial conditions, then the system is only *locally stable*. The stability of an equilibrium point of a system are commonly proven with the Lyapunov's Direct Method; the method is based on the idea that an equilibrium point should be attractive (i.e., have decreasing energy) to be stable.

2.4 Quadrotor Modelling

Quadrotors are VTOL multicopters with exactly four rotors. Although most quadrotors involve body-fixed rotors, attempts of quadrotors with unfixed rotors, such as quadrotors with variable-pitch rotors, have been made [45, 46]. Depending on how the rotors are positioned and whether each of the rotors are fixed or not the dynamics and model of the quadrotor may change. This thesis focuses on the most general type of quadrotor, which has four body fixed rotors each fixed at the cardinal directions (i.e., north, east, south, west) of the body-fixed frame, as shown in Figure 2.6. Each of the rotors are attached on four arms extending from the center of the quadrotor (i.e., the origin of the quadrotor frame). These four rotors rotate either in the same or opposite direction to produce both force and torque, which essentially controls the lift and attitude of the quadrotor. With the exception of some designs, most quadrotors use uni-directional rotor blades which produces an upwards force. The blades are designed slightly differently depending on the direction of rotation, which allows all four rotors to produce a lift force in the same upwards direction even though they rotate in different directions. The four rotors are positioned so that two rotors on opposite ends rotate in one direction, either clockwise or anticlockwise, and two rotors positioned adjacent to each other rotate in opposite directions. The direction of rotation of the four rotors are differently mainly to prevent yaw drift as the quadrotor will lose the ability to produce torque in the yaw direction if all four rotors rotate in the same direction; the other lift force and roll/pitch torques can still be produced even if all rotors rotate in the same direction.

2.4.1 Dynamic Model

The following section describes the quadrotor dynamic model, specifically a quadrotor with four fixed rotors. Borrowing from the frames defined in previous sections, $\mathcal{I} = \{e_x \;\; e_y \;\; e_z\}$ and $\mathcal{A} = \{e_{a,x} \;\; e_{a,y} \;\; e_{a,z}\}$ each define the inertial, or global, frame, in which

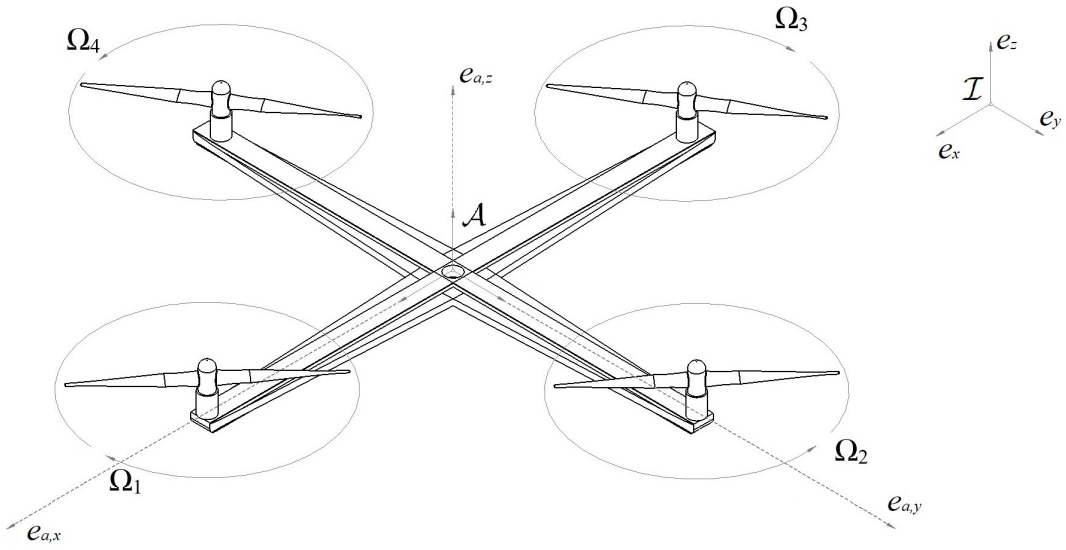


Figure 2.6: Basic Quadrotor Model

the quadrotor resides, and the body, or local, frame of the quadrotor itself. Note that the inertial frame $\mathcal{I}$, in reference to itself, can also represent the canonical basis vectors, ($e_x = [1\ \ 0\ \ 0]^T$, $e_y = [0\ \ 1\ \ 0]^T$, and $e_z = [0\ \ 0\ \ 1]^T$); hence, the three vectors will be used interchangeably, both as the axes of the inertial frame and as the canonical basis vectors, during the remainder of the thesis whenever confusion is unlikely to arise.

In reference to the two defined frames, the dynamic model of the quadrotor, which is based on Tayebi's model [7] original quadrotor model, can be given as follows:

$$\dot{x} = v \tag{2.14}$$

$$m\dot{v} = f_t R_q e_z - g e_z \tag{2.15}$$

$$\dot{q} = \frac{1}{2} q \bar{\omega}_b \tag{2.16}$$

$$I_f \dot{\omega}_b = \tau_a + \omega_b \times I_f \omega_b - G_a \tag{2.17}$$

where $x = [x_1\ \ x_2\ \ x_3]^T$ and $v = [v_1\ \ v_2\ \ v_3]^T$ represent the translational position and

velocity vectors of the quadrotor frame $\mathcal{A}$ with respect to the global frame $\mathcal{I}$. $\dot{*} := \frac{d*}{dt}$ denotes the time derivative of any scalar, vector or quaternion $*$. $q = [q_0 \ q_v^T]^T$ and $\omega_b = [\omega_{b,1} \ \omega_{b,2} \ \omega_{b,3}]^T$ each denote the quaternion attitude and body-fixed angular velocity (i.e., angular velocity with respect to frame $\mathcal{A}$ of the quadrotor. The four vectors/quaternion make up the states of the quadrotor model, which can represent the full six degrees of freedom of the quadrotor (three translation degrees of freedom and three attitude degrees of freedom). Note that $R_q e_z$, though expressed with a rotation matrix, can easily be replaced with the quaternion expression $(qe_zq^{-1})_v$ according to Equation 2.12. m and I_f each denote the mass and inertial matrix of the quadrotor; g denotes the gravitational constant (i.e. $g \approx 9.806 \ m/s^2$). Note that the inertial matrix, I_f, will be assumed diagonal for the remainder of the thesis. The assumption is reasonable as the products of inertia are typically much smaller than the moments of inertia for most well-crafted quadrotors. Gyroscopic torque, $G_a = [G_{a,1} \ G_{a,2} \ G_{a,3}]^T$, is the resulting torque produced due to the rotation the aircraft and the four rotors, which is given by:

$$G_a = \sum_{i=1}^{4} I_r(\omega_b \times e_z)(-1)^{i+1}\Omega_i \tag{2.18}$$

where I_r and $\Omega_i, i \in \{1, 2, 3, 4\}$ each denote the moment of inertia and the angular velocities of the four rotors. f_t and $\tau_a = [\tau_{a,1} \ \tau_{a,2} \ \tau_{a,3}]^T$ are the total thrust and torque. Note that during control both f_t and τ_a represent the generated control thrust and torque inputs designed to achieve stability of the overall system.

Equations 2.14 and 2.15 are the translational position dynamic model, the model are similar to the dynamic model of a system moving in the 3D space. The model is different from land-based vehicles in that it accounts for the gravitational force applied to the quadrotor as it travels in air. Note that the buoyancy force and related torques have been neglected as their influence on the quadrotor dynamics are insignificant. On the other hand, Equations 2.16 and 2.17 are the attitude dynamic model. As with the translational

dynamic model, the two equations includes both the positional information, q, and velocity information, ω_b of the quadrotor attitude. The four equations are coupled with the four different states: x, v, q and ω_b. Equations 2.14 and 2.15 are coupled with the translational velocity vector v, Equations 2.15 and 2.16 are coupled with the attitude quaternion q, or equivalent rotation matrix R_q, and, finally, Equations 2.16 and 2.17 are coupled with the body-fixed angular velocity vector ω_b.

2.4.2 Rotor Velocity Model

The control laws of the overall controller, which will be dealt with extensively in future sections, aims to design the control thrust and torque, f_t and τ_a, that would stabilize the overall quadrotor system. The two control forces work together to achieve a target attitude, which in turns controls the translation position of the quadrotor in 3D space. However, as mentioned before, the actual forces that generate the control thrust and torque are the rotor torques, and the resulting rotor thrusts. To that end, an accurate description of the rotor torques must be established. The following section describes the model of how the rotor forces generate the desired aircraft forces. Both the aircraft thrust, f_t, and aircraft torque, $\tau_a = [\tau_{a,1} \;\; \tau_{a,2} \;\; \tau_{a,3}]^T$ are generated as follows:

$$f_t = \sum_{i=1}^{4} |f_i| = b \sum_{i=1}^{4} {\Omega_i}^2 \tag{2.19}$$

$$\begin{aligned} \tau_{a,1} &= bl({\Omega_2}^2 - {\Omega_4}^2) \\ \tau_{a,2} &= bl({\Omega_1}^2 - {\Omega_3}^2) \\ \tau_{a,3} &= \kappa({\Omega_1}^2 - {\Omega_2}^2 + {\Omega_3}^2 - {\Omega_4}^2). \end{aligned} \tag{2.20}$$

b and κ are two dimensionless parameters; each quantifying the lift and drag of the four rotor blades. l denotes the arm length of the torques created by each of the rotors [7]. Now

let $\mu = [f_t \;\; \tau_{a,1} \;\; \tau_{a,2} \;\; \tau_{a,3}]^T$. Then, Equations 2.19 and 2.20 can be further simplified as

$$\mu = M\Omega^{(2)} \tag{2.21}$$

where $\Omega^{(2)}$ is the element-wise square of the vector Ω and the matrix M is defined as the following:

$$M = \begin{bmatrix} b & b & b & b \\ 0 & bl & 0 & -bl \\ bl & 0 & -bl & 0 \\ \kappa & -\kappa & \kappa & -\kappa \end{bmatrix}.$$

The matrix M well shows the manner in which roll, pitch, yaw torques and the lift force are created with each rotation of the rotors. Roll and pitch torques of the quadrotor can be created as two rotors from opposite ends change angular velocity, while the yaw torque and lift force can be produced as all four rotors change angular velocity. Although the torque model is defined and utilized, a separate control scheme was not designed for the individual torques of the four rotors. The model was mostly used to design the four control angular velocity inputs given the four control lift and torque inputs, which was subsequently fed to the quadrotor under an open loop control environment.

2.4.3 State Space Representation

The dynamic model of the quadrotor can be alternatively represented in state-space representation. As mentioned before, the states of the quadrotor model are consisted of the three vectors, x, v and ω_b, and quaternion, q. Let a 13×1 vector $z = [x^T \;\; v^T \;\; q^T \;\; \omega_b^T]^T$ be the state vector of the overall dynamic model, then the state-space representation of the quadrotor dynamic model can be given as:

$$\dot{z} = f(z, \mu) \tag{2.22}$$

where $f(z,\mu)$ is a set of nonlinear state equations. The state equations $f(z,\mu)$, assuming a diagonal inertial matrix I_f, are given as:

$$f(z,\mu) = \begin{pmatrix} z_4 \\ z_5 \\ z_6 \\ \frac{2\mu_1}{m}(z_7 z_9 + z_8 z_{10}) \\ \frac{2\mu_1}{m}(z_9 z_{10} - z_7 z_8) \\ \frac{\mu_1}{m}(z_7^2 - z_8^2 - z_9^2 + z_{10}^2) - \frac{g}{m} \\ \dots \end{pmatrix} \begin{pmatrix} \frac{-z_8 z_{11} - z_9 z_{12} - z_{10} z_{13}}{2} \\ \frac{z_7 z_{11} + z_9 z_{13} - z_{10} z_{12}}{2} \\ \frac{z_7 z_{12} - z_8 z_{13} + z_{10} z_{12}}{2} \\ \frac{z_7 z_{13} + z_8 z_{12} - z_9 z_{11}}{2} \\ \frac{(I_{f,z} - I_{f,y}) z_{12} z_{13} + \mu_2 - I_r z_{12} \Omega_s}{I_{f,x}} \\ \frac{(I_{f,x} - I_{f,z}) z_{11} z_{13} + \mu_3 + I_r z_{11} \Omega_s}{I_{f,y}} \\ \frac{(I_{f,y} - I_{f,x}) z_{11} z_{12} + \mu_4}{I_{f,z}} \end{pmatrix} \tag{2.23}$$

where $\Omega_s = \sum_{i=1}^{4} (-1)^{i+1} \Omega_i$, derived from Equation 2.18, is the rotor velocity difference, which is the main source of the gyroscopic torque, G_a.

2.4.4 Differential Flatness of Quadrotors

Although the quadrotor is underactuated, the control of quadrotors can be largely simplified due to the differential flatness property of the system. With differentially flat systems, the state and input variables can be represented by a finite number of flat outputs, and the derivatives of the flat outputs, without any additional integration of the dynamic equations [47]. In a differentially flat system, the outputs, or flat outputs, $y \in \mathbb{R}^n$ are:

$$y = \mathcal{Y}(x, u, \dot{u}, ..., u^{(a)})$$

where each of the input variables $x \in \mathbb{R}^m$ and state variables $u \in \mathbb{R}^k$ are as follows:

$$x = \mathcal{X}(y, \dot{y}, ..., y^{(b)})$$
$$u = \mathcal{U}(y, \dot{y}, ..., y^{(c)}).$$

Hence, for underactuated systems the number of inputs can be equal to the number of flat outputs (i.e., $n = m$), which would allow control of all states of the system to be controlled as if the system is fully-actuated. The quadrotor dynamics, like many other underactuated system, inherent such differential flatness with four flat outputs and four control inputs. The four flat outputs include the translational position of the quadrotor within the inertial frame and the yaw of the quadrotor, and the four control inputs are the input forces defined as in Equations 2.19 and 2.20 [48]. Most trajectory tracking controllers of quadrotors utilize the differential flatness property to some extent, with different methods of obtain the four flat outputs of the quadrotor, with some that fully utilize such property to the trajectory designing process of the control [49, 50].

CHAPTER 3
QUADROTOR CONTROL

As discussed during the introduction, the quadrotor can be controlled using various techniques which each have their own advantages and disadvantages. The controller presented in the following chapter is based on nonlinear PID feedback loops. The advantages, compared to other nonlinear techniques, of PID control methods are their simplicity to implement; however, since the control method is based on linear systems, the performance may degrade with high nonlinearities when not dealt with adequately. In addition, PID based methods, when implemented inappropriately, may result in singularities which eventually lead to a failing controller. The proposed controller attempts to control the quadrotor while dealing with singularities and nonlinearities so that the inputs fed into the plant are smooth and continuous. The controller is a dual-loop nonlinear PI/PD controller in which the outer loop controls the overall quadrotor translational position in the 3D space, and the inner loop controls the attitude of the quadrotor within the inertial frame. As shown in Figure 3.1, the control process starts from the controller receiving a reference, or desired, trajectory. The outer translational controller computes the desired thrust vector of the quadrotor, which is subsequently fed to the inner attitude controller to be used to obtain the desired aircraft torques. The desired lift force, on the other hand, are obtained without going through the inner controller. The four input forces, the three desired torques and desired lift, are then fed to plant; and the outputs of the plant are fed back into both the inner and outer controller to complete both feedback loops.

The following sections will be structured as follows. Sections 3.1 and 3.2 will each describe the inner attitude and outer position controllers and present Lyapunov's Stability analysis for both cases. Section 3.3 will discuss the method of which the two control laws are coupled with each other. Section 3.4 discusses different singularity issues and presents

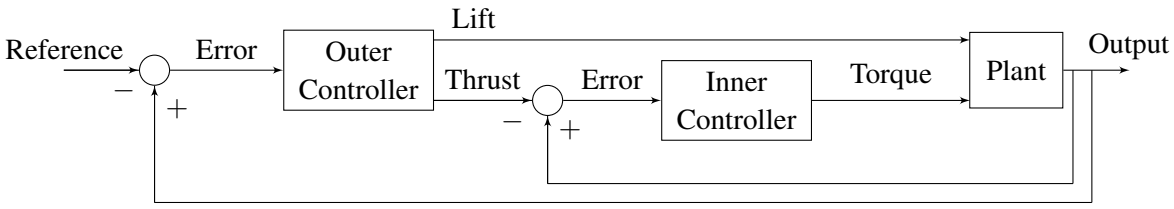


Figure 3.1: Outline of Controller

solutions for the different cases. Section 3.5 conducts an in-depth Lyapunov Stability Analysis after all solutions to achieve global control is appropriately employed. Finally, the chapter will end with Section 3.6 in which a discussion on the final controller will be presented.

3.1 Attitude Control Law

The inner attitude controller is a PD feedback controller, wherein the proportional action is in terms of the attitude error, and the derivative action is in terms of the angular velocity error. To avoid singularities and other issues inherent to the Euler Angle and DCM conventions, the controller utilizes the quaternion convention. Note that when using the quaternion convention most, but not all, singularities are resolved; both the resolved and unresolved singularities will be discussed in Section 3.4. Let q and q_d each denote the actual and desired quaternion attitude, relative to the inertial frame $\mathcal{I}$ in Figure 2.6, of a quadrotor. Note that to achieve global stability, the attitude controller must receive a continuously differentiable and non-singular q_d. Hence, the remainder of this section will assume q_d is smooth and non-singular; the method of how to achieve such q_d will be discussed in future sections. Now let ω_b and ω_d each denote the actual and desired body-fixed angular velocities relative to the body-fixed frame $\mathcal{A}$. Then, the attitude quaternion

error and angular velocity *error* can be expressed as:

$$\tilde{q} = q_d^{-1} q \tag{3.1}$$

$$\tilde{\omega} = \omega_b - \omega_d. \tag{3.2}$$

Recall that to avoid ambiguity $\tilde{q}$ will be constraint by the condition stated in Equation 2.9. With Equation 2.11, the angular velocity error can also be obtained with:

$$\begin{aligned} \tilde{\omega} &= 2(\tilde{q}\dot{\tilde{q}})_v \\ &= 2(q^{-1}\dot{q})_v - 2(q^{-1}\dot{q}_d q_d^{-1} q)_v. \end{aligned}$$

Now let ω_r be a modification of ω_d defined as:

$$\omega_r = \omega_d - \Lambda_q \tilde{q} \tag{3.3}$$

where $\Lambda_q = [0_{3\times 1} \ \lambda_q I_{3\times 3}]$. To avoid confusion ω_d and ω_r will each be referred as the *desired* and *reference* angular velocities, respectively. Finally, the angular velocity error in terms of the reference angular velocities may be obtained with:

$$\tilde{\omega}_r = \omega_b - \omega_r, \tag{3.4}$$

or, by inserting Equation 3.3 into Equation 3.4, alternatively as,

$$\tilde{\omega}_r = \tilde{\omega} + \Lambda_q \tilde{q}. \tag{3.5}$$

The attitude control law derived to achieve global stability is stated as follows:

$$\tau_a = I_f \dot{\omega}_r + \omega_r \times I_f \omega_b + G_a - K_\omega \tilde{\omega}_r - K_q \tilde{q} \tag{3.6}$$

where K_ω and K_q are each gain matrices of the two feedback errors, $\tilde{\omega}_r$ and $\tilde{q}$. K_ω is a 3×3 symmetric positive definite matrix and $K_q = [0_{3\times1} \;\; k_q I_{3\times3}], k_q > 0$. Let the error states of the attitude subsystem be $e_1 = (\tilde{q}, \tilde{\omega}_r) \in \mathbb{R}^7$, then, with the attitude control law defined as Equation 3.6 and the attitude subsystem defined as Equations 2.16 and 2.17, one can derive the following closed-loop error dynamics:

$$\dot{\tilde{q}} = \frac{1}{2}\tilde{q}\,\bar{\tilde{\omega}}_r \tag{3.7}$$

$$I_f\dot{\tilde{\omega}}_r + \tilde{\omega}_r \times I_f\omega_b + K_\omega\tilde{\omega}_r + K_q\tilde{q} = 0. \tag{3.8}$$

The resulting Theorem, which is a generalization of the attitude controller in [7] to time-varying trajectories, proves the global exponential stability of the attitude control error dynamics:

Theorem 3.1. *The attitude closed-loop error system described by Equations* 3.7 *and* 3.8 *resulting from the control law defined by Equation 3.6 applied to the attitude subsystem expressed as Equations* 2.16 *and* 2.17 *is globally exponentially stable at the equilibrium state* $e_1 = \left(\tilde{q} = [1 \;\; 0 \;\; 0 \;\; 0]^T, \tilde{\omega}_r = 0_{3\times1}\right)$ *for a twice continuously differentiable quaternion* q_d *with bounded derivatives.*

Proof. Let the Lyapunov function candidate of the attitude closed-loop error system be:

$$V(\tilde{q}, \tilde{\omega}_r) = \frac{1}{2}\tilde{\omega}_r^T I_f\tilde{\omega}_r + k_q(\tilde{q}-1)^T(\tilde{q}-1) \tag{3.9}$$

The time derivative of V, using Lemma A.1 in Appendix A, is given by:

$$\dot{V}(\tilde{q}, \tilde{\omega}_r) = \tilde{\omega}_r^T I_f\dot{\tilde{\omega}}_r + k_q\bar{\tilde{\omega}}^T\tilde{q}.$$

In view of Equations 3.5 and 3.7, $\dot{V}$ can be simplified as follows:

$$\begin{aligned}\dot{V}(\tilde{q},\tilde{\omega}_r) &= -\tilde{\omega}_r^T(\tilde{\omega}_r \times I_f\omega_b + K_\omega\tilde{\omega}_r + K_q\tilde{q}) + k_q\bar{\tilde{\omega}}^T\tilde{q} \\ &= -\tilde{\omega}_r^T K_\omega\tilde{\omega}_r - \tilde{\omega}_r^T K_q\tilde{q} + k_q\bar{\tilde{\omega}}^T\tilde{q} \\ &= -\tilde{\omega}_r^T K_\omega\tilde{\omega}_r - (\tilde{\omega} + \Lambda_q\tilde{q})^T K_q\tilde{q} + k_q\bar{\tilde{\omega}}^T\tilde{q} \\ &= -\tilde{\omega}_r K_\omega\tilde{\omega}_r - \tilde{q}^T\Lambda_q^T K_q\tilde{q}.\end{aligned}$$

Since Λ_q and K_q are both 3×4 matrices consisted of a zero column vector and a 3×3 diagonal matrix (i.e., $K = [0_{3\times1} \;\; kI_{3\times3}], k > 0$), $\dot{V}(\tilde{q},\tilde{\omega}_r)$ can also be expressed as:

$$\dot{V}(\tilde{q},\tilde{\omega}_r) = -\tilde{\omega}_r K_\omega\tilde{\omega}_r - \lambda_q k_q\tilde{q}_v^T\tilde{q}_v.$$

With the fact that $(\tilde{q}-1)^T(\tilde{q}-1) = 2(1-\tilde{q}_0)$ and Equation 2.8,

$$\tilde{q}_v^T\tilde{q}_v = 1 - \tilde{q}_0^2 = (1-\tilde{q}_0)(1+\tilde{q}_0) = \frac{1+\tilde{q}_0}{2}(\tilde{q}-1)^T(\tilde{q}-1)$$

Hence, $\dot{V}$ can subsequently be given by:

$$\dot{V}(\tilde{q},\tilde{\omega}_r) = -\tilde{\omega}_r K_\omega\tilde{\omega}_r - \lambda_q k_q\frac{1+\tilde{q}_0}{2}(\tilde{q}-1)^T(\tilde{q}-1). \tag{3.10}$$

As a result, $\dot{V}(\tilde{q},\tilde{\omega}_r) \leq -k_1 V$ where $k_1 = \min(\frac{2\lambda_{\min}(K_\omega)}{\lambda_{\max}(I_f)}, \lambda_q)$ and $\lambda_{\min/\max}(\mathbf{m})$ is the minimal or maximum eigenvalue of matrix $\mathbf{m}$. Hence, the attitude closed-loop error system is globally exponentially stable at the equilibrium state [43]. □

Theorem 3.1 can easily be extended to include Λ_q matrices in which the scalar gains are not identical (i.e., $\Lambda_q = [0_{3\times1} \;\; diag(\lambda_{q,1}, \lambda_{q,1}, \lambda_{q,1})]$). In this case, the same Lyapunov equation may be used with $\lambda_{min}(\Lambda_q)$ replacing λ_q.

3.2 Position Control Law

The outer position loop is a PID feedback control, where in the proportional, integral, and derivative action is each in terms of the position error, the integral of the position error and the velocity error. The objective of the outer position control law is to obtain a smooth and continuous thrust vector assuming a twice differentiable continuous reference position trajectory. Let x and x_d each be the *actual* and *desired* linear position of the quadrotor, and v and v_d each be the *actual* and *desired* linear velocity of the quadrotor, all relative to the inertial frame $\mathcal{I}$ (Figure 2.6). Additionally, let $u_d = [u_{d,1} \;\; u_{d,2} \;\; u_{d,3}]^T$ be the *desired* thrust vector, scaled by the aircraft mass, of the quadrotor relative to the inertial frame $\mathcal{I}$. Then, the position control law is given by:

$$u_d = \dot{v}_d - K_v\tilde{v} - K_x\tilde{x} - K_i\int_0^t \tilde{x} + ge_z \tag{3.11}$$

where $\dot{v}_d$ is the *desired* acceleration of the aircraft, $\tilde{x} = x - x_d$ and $\tilde{v} = v - v_d$; x_d and v_d denotes the *desired* linear position and velocity of the aircraft, respectively. The desired linear position $x_d(t)$ is assumed to be smooth with bounded linear velocity $v_d(t)$, acceleration $\dot{v}_d(t)$, and jerk $\ddot{v}_d(t)$.

The gain matrices K_v, K_x, and K_i of the three feedback terms are 3×3, symmetric, positive definite matrices each chosen to make the closed-loop characteristic equation, $|s^3I + s^2K_v + sK_x + K_i|$, Hurwitz (i.e. $Re[\lambda(A)] < 0$, where $Re[*]$ denotes the real part of *). The gains may be chosen either with pole placement or LQR to satisfy the above-mentioned condition. Note that the desired thrust vector obtained from Equation 3.11 is not only passed to the inner attitude control loop but also used to obtain the desired thrust force, which is given by:

$$f_t = m\|u_d\|. \tag{3.12}$$

τ_a and f_t each given by Equation 3.6 and Equation 3.12 makes up the four control input

that is eventually passed to the quadrotor plant. Defining the *error* state of the position subsystem as $e_2 = (w, \tilde{x}, \tilde{v}) \in \mathbb{R}^9$, where $w = \int_0^t \tilde{x}$, the closed-loop error dynamics can then be described by:

$$\dot{w} = \tilde{x} \tag{3.13}$$

$$\dot{\tilde{x}} = \tilde{v} \tag{3.14}$$

$$\dot{\tilde{v}} = -K_v\tilde{v} - K_x\tilde{x} - K_i w + \tilde{u} \tag{3.15}$$

where $\tilde{u} = u - u_d$ is the thrust vector *error*. The actual and desired thrust vectors, u and u_d, are each defined as:

$$u = \frac{f_t}{m} R_q e_z \tag{3.16}$$

$$u_d = \frac{f_t}{m} R_{q_d} e_z. \tag{3.17}$$

The thrust vector error, $\tilde{u}$, in view of Equation 3.16 and Equation 3.17, can also be obtained by:

$$\tilde{u} = \frac{f_t}{m} R_{q_d}(\tilde{R}_q e_z - e_z) \tag{3.18}$$

where $\tilde{R}_q = R_{\tilde{q}}$ is defined in accordance with Equation 2.12 or $\tilde{R}_q = R_{q_d}^T R_q$. Note that the closed-loop error dynamics of the linear position subsystem can also be expressed as:

$$\dot{e}_2 = Ae_2 + B\tilde{u} \tag{3.19}$$

where $A \in \mathbb{R}^{9\times9}$ and $B \in \mathbb{R}^{9\times3}$ denote the state and input matrices, respectively, of the error dynamics state-space representation of Equations 3.13 to 3.15. The two matrices are each given by:

$$A = \begin{bmatrix} 0_{3\times3} & I_{3\times3} & 0_{3\times3} \\ 0_{3\times3} & 0_{3\times3} & I_{3\times3} \\ -K_i & -K_x & -K_v \end{bmatrix} \tag{3.20}$$

$$B = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 \end{bmatrix}^T \tag{3.21}$$

The attitude and position control laws, each desribed in Equation 3.6 and Equation 3.11, are coupled with $\tilde{u}$; with Equation 3.18 and Equation 2.11, $\tilde{q}$, as well as $\tilde{\omega}_r$, can be easily found from the position control law to be used in the attitude control law. However, to prove global stability the attitude error obtained from the conversion must be at least twice continuously differentiable, which is not an easy feature to accomplish. For now, the attitude error $\tilde{q}$ converted from $\tilde{u}$, or q_d obtained from u_d described in Equation 3.11, will be assumed twice continuously differentiable, the method of which this differentiablity is achieved will be discussed in latter sections of this chapter. In addition, Λ_q in Equation 3.3 must be further modified to guarantee global exponential stability of the overall control system. The new modified Λ_q gain to assure global stability is given as:

$$\Lambda_q = (\lambda_q + \lambda_u \left\| u_d \right\|^2)[0_{3\times 1} \;\; I_{3\times 3}] \tag{3.22}$$

where λ_u is an additional positive control gain. Now let the *overall* error state be defined as $e = (e_1, e_2) \in \mathbb{R}^{13}$. The following Theorem shows the global stability of the overall closed-loop dynamic system, stated in Equations 3.7 and 3.8 and Equations 3.13 to 3.15:

Theorem 3.2. *Consider the overall closed-loop error system described by Equations* 3.7 *and* 3.8 *and Equations* 3.13 *to* 3.15 *resulting from the control laws Equation 3.6 and Equation 3.11 with f_t as defined in Equation 3.12 and q_d generated from u_d such that the result q_d is smooth and at least twice continuously differentiable. The overall closed-loop error system is globally exponentially stable at the equilibrium state*

$$e = \left(\tilde{q} = [1 \;\; 0 \;\; 0 \;\; 0]^T, \tilde{\omega}_r = 0_{3\times 1}, w = 0_{3\times 1}, \tilde{x} = 0_{3\times 1}, \tilde{v} = 0_{3\times 1}\right)$$

for a suitably chosen desired trajectory x_d and sufficiently large $\lambda_u k_q$. In particular, $\tilde{\omega}$, $\tilde{v}$,

and $\tilde{x}$ converge to zero exponentially starting from an arbitrary set of initial conditions.

Proof. To prove the Theorem, consider the Lyapunov function candidate:

$$V(\tilde{q}, \tilde{\omega}_r, e_2) = \frac{1}{2}\tilde{\omega}_r^T I_f \tilde{\omega}_r + k_q(\tilde{q}-1)^T(\tilde{q}-1) + e_2^T P e_2. \tag{3.23}$$

where P is a 9×9 matrix that satisfies the Lyapunov equation:

$$A^T P + PA + Q = 0 \tag{3.24}$$

where A is given by Equation 3.20, and Q is an arbitrary symmetric positive definite matrix (Theorem 4.6 and Equation 4.12 in [43]). In view of Equation 3.1 and Equation 3.24, the time derivative of the presented Lyapunov function is then given by:

$$\dot{V}(\tilde{q}, \tilde{\omega}_r, e_2) = -\tilde{\omega}_r^T K_\omega \tilde{\omega}_r - \tilde{q}^T \Lambda_q^T K_q \tilde{q} - e_2^T Q e_2 + 2e_2^T C \tilde{u}$$

where $C = PB$ and B is given by Equation 3.21. Taking Equation 3.22 into consideration, the time derivative of the presented Lyapunov function is then given by:

$$\dot{V}(\tilde{q}, \tilde{\omega}_r, e_2) = -\tilde{\omega}_r^T K_\omega \tilde{\omega}_r - (\lambda_q + \lambda_u \|u_d\|^2) k_q q_v^T q_v - e_2^T Q e_2 + 2e_2^T C \tilde{u}$$

In view of Lemma A.2 (Appendix A), $\dot{V}(\tilde{q}, \tilde{\omega}_r, e_2)$ can be bounded as follows:

$$\begin{aligned}
\dot{V} \leq & -\lambda_{min}(K_\omega)\|\tilde{\omega}_r\|^2 - (\lambda_q + \lambda_u \|u_d\|^2) k_q \|\tilde{q}_v\|^2 - \lambda_{min}(Q)\|e_2\|^2 \\
& + 4c\|e_2\|\,\|u_d\|\,\|\tilde{q}_v\| \\
\leq & -\lambda_{min}(K_\omega)\|\tilde{\omega}_r\|^2 - \lambda_q k_q \|\tilde{q}_v\|^2 \\
& - \begin{bmatrix} \|e_2\| & \|u_d\|\,\|\tilde{q}_v\| \end{bmatrix} \begin{bmatrix} \lambda_{min}(Q) & -2c \\ -2c & \lambda_u k_q \end{bmatrix} \begin{bmatrix} \|e_2\| \\ \|u_d\|\,\|\tilde{q}_v\| \end{bmatrix}
\end{aligned} \tag{3.25}$$

where $c = \|C\|$. It follows that if the following condition:

$$\lambda_{min}(Q)\lambda_u k_q > 4c^2$$

is satisfied, then the 2×2 matrix in the preceding equation is positive definite and consequently $\dot{V} \leq -k_2 V$ for some $k_2 > 0$. All error, therefore, go to zero exponentially for any initial error state e if u_d and q_d are both at least twice continuously differentiable. In addition, since e is bounded and x_d is at least twice differentiable, both $\dot{q}_d$ and ω_d are bounded. Consequently, ω and v are both bounded as well [43]. □

3.3 Obtaining Attitude from Thrust Vector

There are multiple methods of which u_d can be used to find the desired attitude of the quadrotor, assuming a smooth and well-defined normalized vector of the thrust vector, u_d. All methods are possible due to the inherent differential flatness property of the quadrotor as discussed in subsection 2.4.4. For the remainder of the section let $\hat{u}_d$ be the normalized vector of u_d (i.e., $\hat{u}_d = \frac{u_d}{\|u_d\|}$). Each method has each of their own pros and cons, this section will discuss three methods each based on the Euler Angle, DCM, and quaternion convention. Each of the three conventions, and conversion methods, are related to some extend; however, each of the conversion methods display different characteristics which would make them useful for different applications.

3.3.1 $\hat{u}_d$ to $\alpha\beta$

Viewing the normalized desired thrust vector, $\hat{u}_d$, as the body-fixed z-axis of the desired rotation frame, a rotation from the global z-axis e_z to $\hat{u}_d$ in the Euler Angle con-

vention, specifically the roll and pitch angles, can be obtained by:

$$\alpha_d = -\operatorname{atan2}\left(\hat{u}_{d,2}, \hat{u}_{d,3}\right)$$
$$\beta_d = \operatorname{atan2}\left(\hat{u}_{d,1}, \sqrt{\hat{u}_{d,2}^2 + \hat{u}_{d,3}^2}\right)$$

where α_d and β_d are the desired roll and pitch angles derived from a u_d described in Equation 3.11. As mentioned before, though the method based on Euler Angle is simple and intuitive, it suffers from singularity. The singularity occurs when the second angle of rotation, the pitch angle, reaches $\frac{\pi}{2}$, which is right in the middle of the 3D rotation frame making it undesirable to use for a global attitude controller.

3.3.2 $\hat{u}_d$ to R_d

The second method also views the desired thrust vector $\hat{u}_d$ as the $e_{a,z}$ axis, but instead of directly obtaining the rotation angles, it involves attaching arbitrary $e_{a,x}$ and $e_{a,y}$ axis to complete the frame and finding a rotation matrix from e_z to $e_{a,z}$ that corresponds to the frame. The method consists of 3 consecutive cross product operations to find the arbitrary $e_{a,x}$ and $e_{a,y}$ axes. The three axes are given by,

$$u_{d,y} = \frac{\hat{u}_d \times e_x}{\|\hat{u}_d \times e_x\|}$$
$$u_{d,x} = \frac{u_{d,y} \times \hat{u}_d}{\|u_{d,y} \times \hat{u}_d\|}$$
$$u_{d,z} = \hat{u}_d$$

where $e_x = [1 \;\; 0 \;\; 0]^T$ denotes the canonical basis vector. Then desired rotation matrix, R_d, can be given by,

$$R_d = \begin{bmatrix} u_{d,x} & u_{d,y} & u_{d,z} \end{bmatrix}.$$

Note that this method, similar to Euler Angle method, introduces singularities, specifically when $\hat{u}_d$ and e_x are parallel (i.e., $\hat{u}_d = [\pm 1 \;\; 0 \;\; 0]^T$), which again makes it unsuitable for

globally stable controllers.

3.3.3 $\hat{u}_d$ to q_d

The third method converts $\hat{u}_d$ directly to q_d which, at the same time, guarantees that q_d is not-singular as long as $\hat{u}_d \neq [0 \;\; 0 \;\; -1]^T$. With Equation 2.13, q_d, which is essentially the minimal rotation from e_z to $\hat{u}_d$, can be obtained with the following quaternion:

$$q_d = \frac{1}{\sigma_d} \begin{bmatrix} 1 + \hat{u}_{d,3} \\ -\hat{u}_{d,2} \\ \hat{u}_{d,1} \\ 0 \end{bmatrix} \tag{3.26}$$

where $\sigma_d = \sqrt{1 + 2\hat{u}_{d,3} + \|\hat{u}_d\|^2}$ is the normalization factor to produce a unit quaternion q_d. Note that the quaternion method is not completely singular-free; however, it is much simpler to design with as the singular point has been reduced to a single point which is the furthest away from equilibrium (i.e., $\hat{u}_d = [0 \;\; 0 \;\; 1]^T$); other methods have two singular points which are both right in the middle of the rotation frame.

3.3.4 Comparison

Figure 3.2 shows the locations of the singularities for each of the three methods; the singularities present in the Euler Angles, and DCM, based methods are shown in red, and the singularities present in the quaternion method are shown in blue.

To demonstrate each of the singularities a $\hat{u}_d$ vector, which represents the thrust vector when the quadrotor rotates $360°$, was used to find the corresponding minimal rotation with each of the conversion methods. The $\hat{u}_d$ vector used for this simple test can be seen in Figure 3.3. This particular trajectory was chosen as it passes through the all three singularities shown in Figure 3.2. Each of the methods, described in Sections 3.3.1 to 3.3.3, was used to convert $\hat{u}_d$ to its corresponding attitude representations. As displayed in Figures 3.4a and 3.4b, after $\hat{u}_d$ passes through the singular points (i.e., $\hat{u}_{d,3} < 0$), the attitude

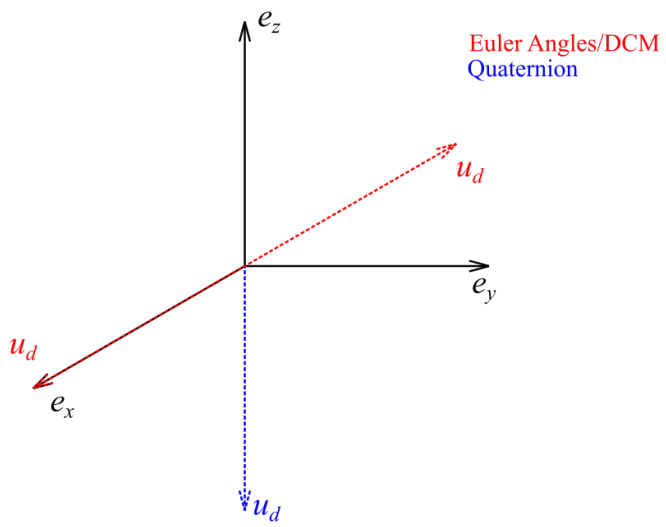


Figure 3.2: Singularity Locations

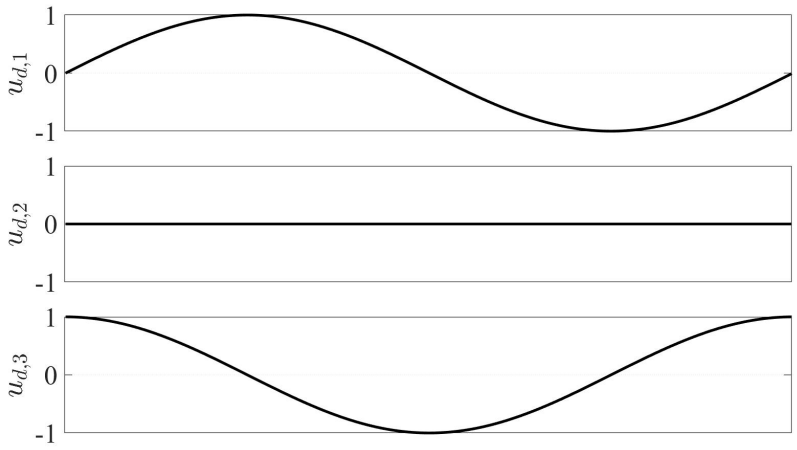
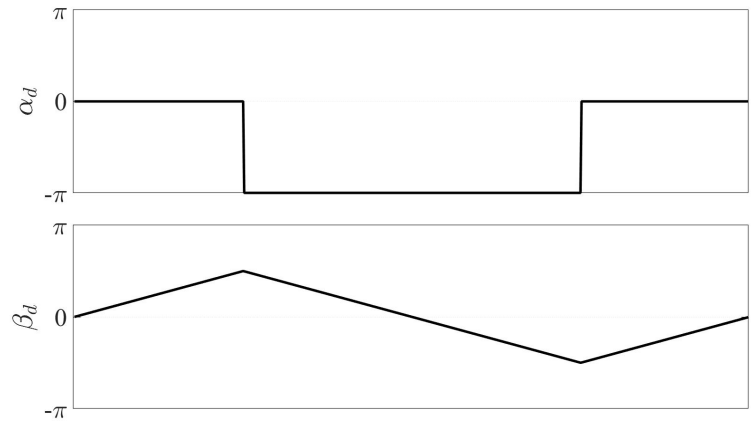
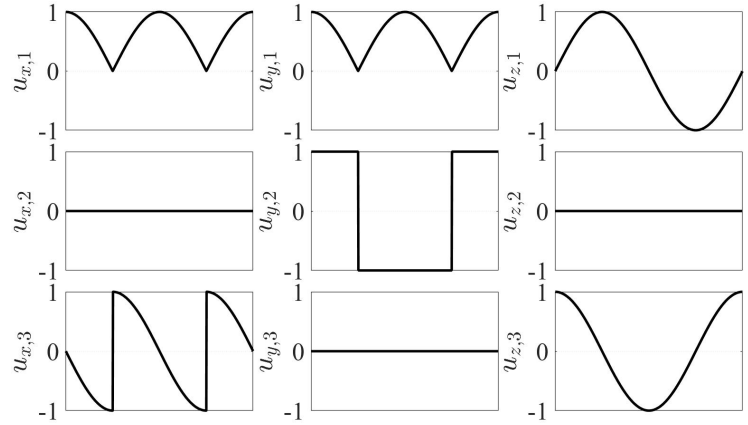


Figure 3.3: $\hat{u}_d$ Trajectory

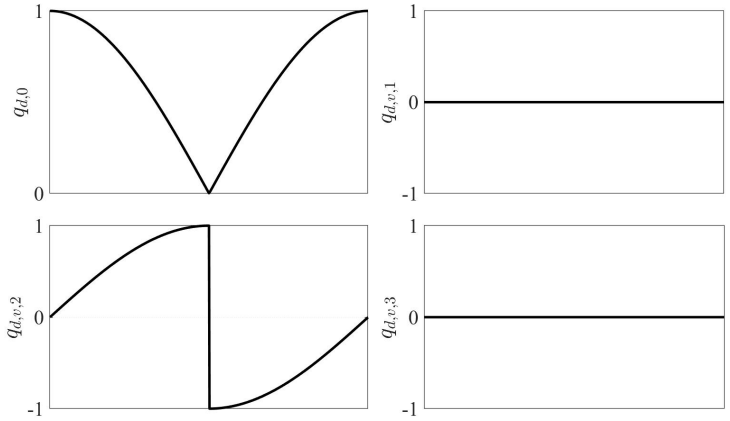
of $\hat{u}_d$ in Euler Angles and DCM representations are not continuous. This discontinuity is mainly due to the fact that part of the attitude information is included in the rotation around the e_z axis, which the two methods, do not consider. Hence, the attitude representations of $\hat{u}_d$ found with the two methods experience a jump from one attitude point to another. The methods can certainly be modified to account for the two singularities. For instance, both methods can account for the desired axis rotation, obtained with $\hat{r} = \hat{u}_d \times e_z$, which will allow each method to to avoid the singular points. Even so, those methods will only lead to the same singularity shown in the third quaternion based method as it is difficult to obtain



(a) $\hat{u}_d \rightarrow \alpha_d, \beta_d$



(b) $\hat{u}_d \rightarrow R_d$



(c) $\hat{u}_d \rightarrow q_d$

Figure 3.4: $\hat{u}_d$ Conversion Results

the desired axis of rotation when $\hat{u}_d$ is parallel with e_z (i.e., $\hat{u}_d = [0 \;\; 0 \;\; -1]^T$) as there are numerous axes of rotation that can rotate e_z to $\hat{u}_d = [0 \;\; 0 \;\; -1]^T$ with a rotation angle of $\pm\pi$.

Figure 3.4c shows that the quaternion based method accurately describes the attitude of the quadrotor before and after the singular point. MATLAB makes it difficult to actually see the singularity which occurs when $\hat{u}_d = [0 \;\; 0 \;\; -1]^T$ as it has very high precision. However, typical micro-controllers, used during implementation, usually has much lower precision than MATLAB, which makes the singularity impact the control more significantly. Note that although the singularity evident in Figure 3.4c may seem like a *jump* singularity, it is actually a *removable* singularity. The jump only exists because the quaternion is forced to be subject to the constraint Equation 2.9 to avoid ambiguity; in fact, the resulting desired attitude does not actually jump. On the other hand, the singularities present in Figures 3.4a and 3.4b are *jump* singularities; both resulting Euler Angles, and DCMs, does show a jump from one attitude to another when passing through the singularity. Moreover, literature does provide several methods of solving the jump singularities present in Euler angles [51]; however, the different solutions eventually leads to the removable singularity that exists in the quaternion-based conversion. Hence, it is more advantageous to use the quaternion-based method as removable singularities are much easier to deal with compared to jump singularities, and given the fact that the jump singularity inevitably occurs for the methods based on all three attitude conventions.

3.3.5 Yaw Control

The three method of finding corresponding rotations from $\hat{u}_d$ each presented in Sections 3.3.1 to 3.3.3 does not account for the aircraft yaw. The method based on Euler angles only calculates the desired roll and pitch angles, and both the methods based on DCM and quaternions assumes zero yaw. In fact, in a dual-loop control configuration, it is common to fix the yaw at a certain angle. In the Euler angle based method, the yaw can be fixed by setting γ_d to any constant angle. The other two methods can incorporate yaw by multiplying a rotation matrix, or quaternion, that is equivalent to the amount of yaw

rotation (i.e.,

$$R_d = R_{d,t} R_{\gamma_d}$$

$$q_d = q_{d,t} q_{\gamma_d}$$

where R_d and q_d each represent the final desired rotations, and $R_{d,t}$ and $q_{d,t}$ denote the rotations without yaw each obtained with the methods described in Sections 3.3.2 and 3.3.3).

To this end, let $\hat{\gamma}$ represent the some rotation angle that is associated with the rotation around the e_z axis, and $\hat{\phi}$ denote the corresponding rotation angles around an axis on the $e_x e_y$ plane. Then the corresponding quaternions, $q_{\hat{\gamma}}$ and $q_{\hat{\phi}}$, can be given as:

$$q_{\hat{\gamma}} = \begin{bmatrix} \cos\left(\frac{\hat{\gamma}}{2}\right) \\ 0 \\ 0 \\ \sin\left(\frac{\hat{\gamma}}{2}\right) \end{bmatrix} \quad q_{\hat{\phi}} = \begin{bmatrix} \cos\left(\frac{\hat{\phi}}{2}\right) \\ \hat{r}_1 \sin\left(\frac{\hat{\phi}}{2}\right) \\ \hat{r}_2 \sin\left(\frac{\hat{\phi}}{2}\right) \\ 0 \end{bmatrix} \tag{3.27}$$

where $\hat{r} = [\hat{r}_1 \;\; \hat{r}_2 \;\; 0]^T$ is a rotation axis on the $e_x e_y$ plane. $\hat{\gamma}$ can be extracted from the quaternion $q = [q_0 \;\; q_v]^T$ from the following relationship [52]:

$$\hat{\gamma} = 2 \operatorname{atan2}(q_{v,3}, q_0). \tag{3.28}$$

The following Proposition shows that any quaternion q can be separate into the product of two quaternions, a quaternion with an angle of $\hat{\gamma}$ around the e_z axis and a $\hat{\gamma}$ free quaternion with an angle $\hat{\phi}$ around an axis of rotation with zero z-component (i.e., $\hat{r} = [\hat{r}_1 \;\; \hat{r}_2 \;\; 0]^T$).

Proposition 3.1. *Let $q_{\hat{\gamma}}$ and $q_{\hat{\phi}}$ each be unit quaternions as defined in Equation 3.27, where $\hat{\gamma}$ can be obtained with Equation 3.28. If the rotation axis and angle, $\hat{r}$ and ϕ, of $q_{\hat{\phi}}$ are chosen to correspond a rotation from $R_q^T e_z$ to e_z. Then, the three quaternions, q, q_γ and q_ϕ, satisfies the following relationship:*

$$q = q_{\hat{\phi}} q_{\hat{\gamma}}. \tag{3.29}$$

Proof. Let $q_{\hat{\phi}}$ be the minimal quaternion that rotates e_z to u as described in Equation 2.13, where $u = R_q e_z$ can be found with the following conversion from quaternion to rotation matrix:

$$R_q = \begin{bmatrix} 2(q_0^2 + q_{v,1}^2) - 1 & 2(q_{v,1}q_{v,2} - q_0 q_{v,3}) & 2(q_{v,1}q_{v,3} + q_0 q_{v,2}) \\ 2(q_{v,1}q_{v,2} + q_0 q_{v,3}) & 2(q_0^2 + q_{v,2}^2) - 1 & 2(q_{v,2}q_{v,3} - q_0 q_{v,1}) \\ 2(q_{v,1}q_{v,3} - q_0 q_{v,2}) & 2(q_{v,2}q_{v,3} + q_0 q_{v,1}) & 2(q_0^2 + q_{v,3}^2) - 1 \end{bmatrix}$$

Then,

$$R_q e_z = \begin{bmatrix} 2(q_{v,1}q_{v,3} + q_0 q_{v,2}) \\ 2(q_{v,2}q_{v,3} - q_0 q_{v,1}) \\ 2(q_0^2 + q_{v,3}^2) - 1 \end{bmatrix},$$

and $q_{\hat{\phi}}$, with Equation 2.13, can be given as:

$$q_{\hat{\phi}} = \begin{bmatrix} 2(q_0^2 + q_{v,3}^2) \\ -2(q_{v,2}q_{v,3} - q_0 q_{v,1}) \\ 2(q_{v,1}q_{v,3} + q_0 q_{v,2}) \\ 0. \end{bmatrix} \tag{3.30}$$

Additionally, $q_{\hat{\gamma}}$, with $\hat{\gamma}$ defined as Equation 3.28, can be obtained with the following expression:

$$q_{\hat{\gamma}} = \begin{bmatrix} \cos\left(\text{atan2}\left(q_{v,3}, q_0\right)\right) \\ 0 \\ 0 \\ \sin\left(\text{atan2}\left(q_{v,3}, q_0\right)\right) \end{bmatrix} = \begin{bmatrix} \frac{q_0}{\sqrt{q_0^2 + q_{v,3}^2}} \\ 0 \\ 0 \\ \frac{q_3}{\sqrt{q_0^2 + q_{v,3}^2}} \end{bmatrix} \tag{3.31}$$

With Equations 3.30 and 3.31, $q_{\hat{\phi}}q_{\hat{\gamma}}$ can be solved as follows:

$$
\begin{aligned}
q_{\hat{\phi}}q_{\hat{\gamma}} &= \begin{bmatrix} 2(q_0^2+q_{v,3}^2) \\ -2(q_{v,2}q_{v,3}-q_0q_{v,1}) \\ 2(q_{v,1}q_{v,3}+q_0q_{v,2}) \\ 0. \end{bmatrix} \begin{bmatrix} \frac{q_0}{\sqrt{q_0^2+q_{v,3}^2}} \\ 0 \\ 0 \\ \frac{q_3}{\sqrt{q_0^2+q_{v,3}^2}} \end{bmatrix} \\
&= \begin{bmatrix} \frac{2q_0(q_0^2+q_{v,3}^2)}{\sqrt{q_0^2+q_{v,3}^2}} \\ \frac{-2q_0(q_{v,2}q_{v,3}-q_0q_{v,1})+2q_{v,3}(q_{v,1}q_{v,3}+q_0q_{v,2})}{\sqrt{q_0^2+q_{v,3}^2}} \\ \frac{2q_0(q_{v,1}q_{v,3}+q_0q_{v,2})+2q_{v,3}(q_{v,2}q_{v,3}-q_0q_{v,1})}{\sqrt{q_0^2+q_{v,3}^2}} \\ \frac{2q_{v,3}(q_0^2+q_{v,3}^2)}{\sqrt{q_0^2+q_{v,3}^2}} \end{bmatrix} \\
&= \begin{bmatrix} \frac{2q_0(q_0^2+q_{v,3}^2)}{\sqrt{q_0^2+q_{v,3}^2}} \\ \frac{2q_{v,1}(q_0^2+q_{v,3}^2)}{\sqrt{q_0^2+q_{v,3}^2}} \\ \frac{2q_{v,2}(q_0^2+q_{v,3}^2)}{\sqrt{q_0^2+q_{v,3}^2}} \\ \frac{2q_{v,3}(q_0^2+q_{v,3}^2)}{\sqrt{q_0^2+q_{v,3}^2}} \end{bmatrix} = \frac{1}{\sigma_\gamma} \begin{bmatrix} q_0 \\ q_{v,1} \\ q_{v,2} \\ q_{v,3} \end{bmatrix}
\end{aligned}
$$

where $\sigma_\gamma = 2\sqrt{q_0^2+q_{v.3}^2}$ is the normalization factor of $q_{\hat{\phi}}q_{\hat{\gamma}}$. Hence, the resulting quaternion is equivalent to the original quaternion once normalized (i.e., $q = \frac{q_{\hat{\phi}}q_{\hat{\gamma}}}{\sigma_\gamma}$). □

The controller will incorporate $q_{\hat{\gamma}}$ so that the the tilt rotation, $q_{\hat{\phi}}$ is controlled by u_d, while $\hat{\gamma}$ is controlled separately in which the $\hat{\gamma}$ extracted from the actual attitude quaternion q is fixed at a constant angle $\hat{\gamma}_d$. Note that, while $\hat{\gamma}$ is related to the γ, the traditional definition of yaw, a certain extent, $\hat{\gamma} \neq \gamma$. Hence, some yaw, defined in the traditional sense, may be introduced during control. The decomposition of quaternion q into $q_{\hat{\gamma}}$ and $q_{\hat{\phi}}$ allows the two quaternions to be controlled separately without any singularity if $q_0 \neq q_{v,3}$ as $\mathrm{atan2}(0,0)$ is undefined, which occurs when the attitude is completely inverted.

3.3.6 *Alternative Approach of Finding $\tilde{q}$*

There are two different ways to find $\tilde{q}$. The first method involves obtaining q_d then subsequently obtaining $\tilde{q}$ with Equation 3.1. The second method involves finding $\tilde{q}$ directly from $\check{u}_d$ by modifying the method described in Equation 3.26. Let $\check{u}_d$ be the normalized *desired* thrust vector relative to the body-fixed frame $\mathcal{A}$, $\check{u}_d$ can be given by,

$$\check{u}_d = R_q^T \frac{u_d}{\|u_d\|} \tag{3.32}$$

where R_q is the DCM representation of the actual quadrotor attitude, or the body-fixed frame of the quadrotor. Given $\check{u}_d$ from Equation 3.32, $\tilde{q}$ can be found as follows:

$$\tilde{q} = \frac{1}{\sigma_d} \begin{bmatrix} 1 + \check{u}_{d,3} \\ \check{u}_{d,2} \\ -\check{u}_{d,1} \\ 0 \end{bmatrix} \tag{3.33}$$

where, as before, $\sigma_d = \sqrt{1 + 2\check{u}_{d,3} + \|\check{u}_d\|^2}$ is the normalization factor to produce a unit quaternion $\tilde{q}$. Note that, unlike the method of finding q_d first then obtaining $\tilde{q}$, the second method of finding $\tilde{q}$ directly from u_d results in the minimal quaternion rotation from e_z to $-R_q^T \frac{u_d}{\|u_d\|}$, or the rotation from $R_q^T \frac{u_d}{\|u_d\|}$ to e_z. The advantage of finding $\tilde{q}$ using the second method is in that the second method requires the $R_q^T \frac{u_d}{\|u_d\|} \neq [0 \;\; 0 \;\; -1]^T$, instead of $\frac{u_d}{\|u_d\|} \neq [0 \;\; 0 \;\; -1]^T$, to be singular. This singularity is much more difficult to achieve as it involves u_d being opposite and parallel to u, and, assuming continuous stable and accurate control with small sampling times, the error between the actual and desired thrust vectors does not get as large. On the other hand, $u_d = [0 \;\; 0 \;\; -1]^T$ can be much more easily reached depending on the trajectory of the control. For instance, the u_d trajectory shown in Figure 3.3 passes through the singularity. Also, the alternative formulation is particularly appealing when the angle between u and u_d is acute making the computation of $q_{\min}(e_z, \check{u}_d)$

numerically more robust than the original approach.

If $\tilde{q}$ is obtained with Equation 3.33, the yaw component of $\tilde{q}$ is now assumed to be zero (i.e., the fourth component of $\tilde{q}$ remains 0 for all desired trajectories); hence, according to Equation 3.6, the yaw control torque $\tau_{a,3}$ only includes the 3 nonlinear compensation terms and the angular velocity control term. Stable control of the quadrotor is still possible without $\tilde{q}_{v,3}$ in the control torques; however, robust control is difficult to achieve, especially with the introduction of disturbances during large attitude changes. More importantly, the control method does not rule out the possibility of constant velocity spin in the yaw direction. Hence, as with the previous method of finding q_d, yaw control must be added separately. This can be done by finding the overall quaternion attitude error as:

$$\tilde{q} = \tilde{q}_{\hat{\gamma}} \tilde{q}_{\phi} \tag{3.34}$$

where $\tilde{q}_{\hat{\gamma}}$ represents the error between the $\hat{\gamma}$ angles each associated with the desired and actual quaternion defined by Equation 3.28, and $\tilde{q}_{\phi}$ represents the minimal quaternion rotation from e_z to $-R_q^T \frac{u_d}{\|u_d\|}$ obtained with Equation 3.33.

3.4 Singularities

The following section will discuss the different singularities that were briefly introduced in Sections 3.1 to 3.3. Even with the use of quaternions, which is in itself not completely singularity-free, some major singularities must be addressed to truly achieve global stability in the sense that all attitudes can be achieved. Some singularities are not removable; hence, will be avoided with smoothing functions. Such smoothing functions will only be used to eliminate singularities that rarely occur. Other, more easily removed singularities, will be eliminated by approximations.

3.4.1 *Vanishing Thrust Vector*

A vanishing thrust vector occurs when $u_d(t) = [0\ 0\ 0]^T$. According to Equation 3.11, this occurs when the right side of the equation is exactly zero. This rarely occurs as $u_d = 0_{3\times1}$ represents a free-falling motion, and, most controllers, are implemented so that such motion does not occur. Nevertheless, it is still critical for the controller to be able to handle the singularity when it does occur. To this end, let $0 < \epsilon < \frac{1}{2}$ be a small positive scalar and let $\sigma : \mathbb{R}^+ \to \mathbb{R}^+$ be a C^2 switching function given by:.

$$\sigma(\xi) = \begin{cases} \frac{1}{2}\epsilon\Big(1 + \xi^2\epsilon^{-2}\Big) & \text{if } 0 \leq \xi \leq \epsilon \\ \xi & \text{if } \xi \geq \epsilon \end{cases}. \tag{3.35}$$

Figure 3.5 shows the resulting σ values as ϵ varies from 0.05 to 0.4. It can be seen that each $\sigma(.)$ is smooth with boundary conditions $\sigma(0) = \frac{\epsilon}{2}$ and $\sigma(\xi) = \xi$ for $\xi \geq \epsilon$. This smoothing function can be used to modify Equation 3.32 into:

$$\hat{u}_d = \frac{u_d}{\sigma(\|u_d\|)}. \tag{3.36}$$

$\sigma(\xi)$ is only being applied to $\|u_d\|$; therefore, $u_d(t)$ can still be a zero vector. In this case, according to Equations 3.12 and 3.26, $u_d = 0_{3\times1}$ leads to $f_t = 0$ and $q_d = [1\ 0\ 0\ 0]^T$ without singularity.

3.4.2 $\hat{u}_d = [0\ 0\ -1]^T$

q_d and $\tilde{q}$, each obtained with Equations 3.26 and 3.33, are both minimal, $\hat{\gamma}$ free, quaternions, which rotates a unit vector v to another unit vector w. In particular, q_d represents the minimal rotation from e_z to $\hat{u}_d$ (i.e., $q_{min}(e_z, \hat{u}_d)$), while $\tilde{q}$ from Equation 3.33 represents the minimal rotation from e_z to $-R_q^T\hat{u}_d$ (i.e., $q_{min}(e_z, -R_q^T\hat{u}_d)$). As mentioned before, singularity also occurs when $\hat{u}_d = \frac{u_d}{\|u_d\|}$ or $-R_q^T\hat{u}_d$ is parallel and opposite to e_z.

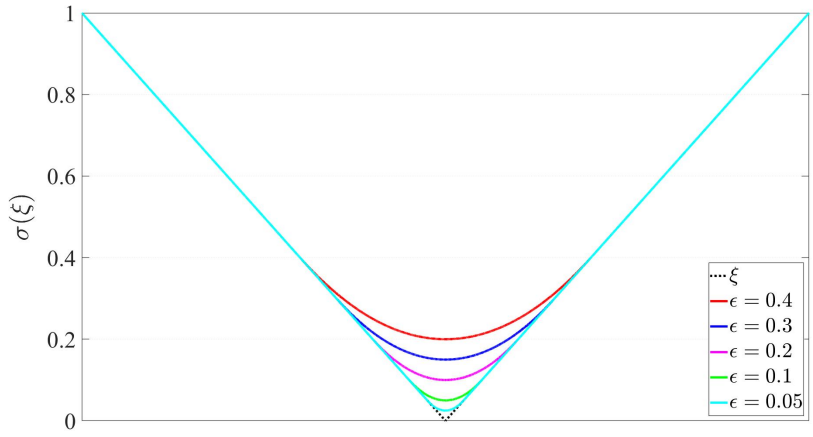


Figure 3.5: σ with Varying ϵ

For both cases, the rotation is undefined and singular as there are numerous rotations that can get one unit vector to another unit vector if the two vectors are parallel and opposite; in fact, a rotation of π with any axes that lies on the $e_x e_y$ plane can accomplish this. To this end, the following Proposition shows that $q_{min}(\mathbf{v}, \mathbf{w})$ can be obtained for any arbitrary rotation that rotates $\mathbf{v}$ to $\mathbf{w}$ without running into any singularities, even when $\mathbf{v} = -\mathbf{w}$.

Proposition 3.2. *Let* $\mathbf{v}$ *be a fixed unit vector in* $\mathbb{R}^3$, $\mathbf{w}(t)$ *a* C^2 *vector with* $\|\mathbf{w}(t)\| \leq 1$, *and* $\mathbf{u}(t)$ *a* C^2 *unit vector satisfying either* $\mathbf{v}^T\mathbf{u}(t) = 0$ *or* $\mathbf{v}^T\mathbf{u}(t) = \mathbf{w}(t)^T\mathbf{u}(t)$. *For any* $0 < \delta \leq 1$ *there exists a* C^2 *unit quaternion* $q_\delta(\mathbf{u}(t), \mathbf{v}, \mathbf{w}(t))$ *such that*

$$q_\delta(\mathbf{u}, \mathbf{v}, \mathbf{w}) = \begin{cases} q_{\min}(\mathbf{v}, \mathbf{w}), & \mathbf{w}^T\mathbf{v} \geq -1 + \delta \\ \bar{\mathbf{u}}, & \mathbf{w}^T\mathbf{v} = -1 \end{cases}$$

and $\bar{\mathbf{w}} = q_\delta(\mathbf{u}, \mathbf{v}, \mathbf{w})\bar{\mathbf{v}}q_\delta(\mathbf{u}, \mathbf{v}, \mathbf{w})^{-1}$ *if* $\|\mathbf{w}\| = 1$.

Proof. First we shall generate a unit quaternion $q = [q_0 \ q_v^T]^T$ that rotates $\mathbf{v}$ to $\mathbf{w}$. Let $\hat{\phi} = \cos^{-1}(\mathbf{w}^T\mathbf{v})$ and $\zeta = 0$ if $\mathbf{v}^T\mathbf{u}(t) = \mathbf{w}(t)^T\mathbf{u}(t)$ and $\zeta = \frac{\mathbf{u}^T\mathbf{w}}{1-\cos(\hat{\phi})}$ otherwise (i.e.,

$\mathbf{v}^T\mathbf{u}(t) = 0$). Then, it can be seen that

$$q_0 = \frac{\sqrt{\cos^2(\frac{\hat{\phi}}{2}) - \sin^2(\frac{\hat{\phi}}{2})(\zeta^2 + (\mathbf{v}^T\mathbf{u})^2)}}{\sqrt{1-(\mathbf{v}^T\mathbf{u})^2}}$$
$$q_v = \frac{\sin(\frac{\hat{\phi}}{2})}{\sqrt{1-(\mathbf{v}^T\mathbf{u})^2}}(\mathbf{u} + \zeta\mathbf{v})$$

are smooth functions of $\mathbf{u}$ and $\mathbf{w}$ everywhere except possibly at $\mathbf{w} = \mathbf{v}$ and the resulting quaternion q rotates $\mathbf{v}$ to $\mathbf{w}$ if $\|\mathbf{w}\| = 1$: $\overline{\mathbf{w}} = q\overline{\mathbf{v}}q^{-1}$. From the expressions for q_0 and q_v, it also can be seen that

$$\frac{1+\mathbf{w}^T\mathbf{v}}{2} = \cos^2\left(\frac{\hat{\phi}}{2}\right) = (q_0^2 + (q_v^T\mathbf{v})^2)(1-(\mathbf{v}^T\mathbf{u})^2)$$

If $\mathbf{v}^T\mathbf{u} \neq 0$, then $\mathbf{u}^T\mathbf{v} = \mathbf{u}^T\mathbf{w}$ implying that

$$\mathbf{u}^T\mathbf{v} = \mathbf{u}^T\tilde{\mathbf{w}}$$

where $\tilde{\mathbf{w}} = \frac{(\mathbf{v}+\mathbf{w})}{2}$ and $\|\tilde{\mathbf{w}}\|^2 = \frac{(1+\mathbf{w}^T\mathbf{v})}{2}$. Consequently, $(\mathbf{u}^T\mathbf{v})^2 \leq \frac{(1+\mathbf{w}^T\mathbf{v})}{2}$ and if $1 + \mathbf{w}^T\mathbf{v} \leq \delta$, then $q_0 \leq \epsilon$ and $|\mathbf{v}^T q_v| \leq \epsilon_\delta$ where $\epsilon_\delta := \sqrt{\frac{\delta}{2(1-\delta)}}$. To formulate $q_\delta(\mathbf{u}, \mathbf{v}, \mathbf{w})$, let $\sigma : \mathbb{R}^+ \to \mathbb{R}^+$ be a C^2 switching function given by Equation 3.35 with $\epsilon \leq \epsilon_\delta$, and define $q_{\hat{\gamma}} = [\cos(\frac{\hat{\gamma}}{2}) \;\; \sin(\frac{\hat{\gamma}}{2})\mathbf{v}^T]^T$ with

$$\sin\left(\frac{\hat{\gamma}}{2}\right) = \frac{\mathbf{v}^T q_v}{\sqrt{\sigma(q_0)^2 + (\mathbf{v}^T q_v)^2}}$$
$$\cos\left(\frac{\hat{\gamma}}{2}\right) = \frac{\sigma(q_0)}{\sqrt{\sigma(q_0)^2 + (\mathbf{v}^T q_v)^2}}$$

We are now in position to completely specify

$$q_\delta(\mathbf{u}, \mathbf{v}, \mathbf{w}) = \begin{cases} q_{\min}(\mathbf{v}, \mathbf{w}) & \mathbf{w}^T\mathbf{v} \geq -1 + \delta \\ q q_{\hat{\gamma}}^{-1}, & \mathbf{w}^T\mathbf{v} < -1 + \delta \end{cases}$$

with

$$qq_{\hat{\gamma}}^{-1} = \begin{bmatrix} q_0 \cos(\frac{\hat{\gamma}}{2}) + \sin(\frac{\hat{\gamma}}{2}) q_v^T \mathbf{v} \\ \cos(\frac{\hat{\gamma}}{2}) q_v - \sin(\frac{\hat{\gamma}}{2})(q_0 \mathbf{v} + q_v \times \mathbf{v}) \end{bmatrix}$$

We first prove that $qq_{\hat{\gamma}}^{-1} = q_{\min}(\mathbf{v}, \mathbf{w})$ on the open subset $\mathcal{N} = \{\mathbf{w} \in \mathcal{S} : -1 + \delta < \mathbf{w}^T \mathbf{v} < 1\}$ of the closed unit sphere. For all $\mathbf{w} \in \mathcal{N}$, $\sigma(q_0) = q_0$, $\cos(\frac{\hat{\phi}}{2}) = \sqrt{q_0^2 + (\mathbf{v}^T q_v)^2}$, and

$$\begin{aligned} \sin\left(\frac{\hat{\gamma}}{2}\right) &= \mathbf{v}^T q_v \sec\left(\frac{\hat{\phi}}{2}\right) \\ \cos\left(\frac{\hat{\gamma}}{2}\right) &= q_0 \sec\left(\frac{\hat{\phi}}{2}\right) \end{aligned} \tag{3.37}$$

The equality of the scalar component of $qq_{\hat{\gamma}}^{-1}$ and $q_{\min}(\mathbf{v}, \mathbf{w})$, which is $\cos(\frac{\hat{\phi}}{2})$, follows directly from $q_0 = \cos(\frac{\hat{\gamma}}{2}) \cos(\frac{\hat{\phi}}{2})$ and $q_v^T \mathbf{v} = \sin(\frac{\hat{\gamma}}{2}) \cos(\frac{\hat{\phi}}{2})$. To prove the equality of the vector components, it is sufficient to establish that the vector component of $qq_{\hat{\gamma}}^{-1}$ is orthogonal to both v and w:

$$\mathbf{w}^T (qq_{\hat{\gamma}}^{-1})_v = \mathbf{v}^T (qq_{\hat{\gamma}}^{-1})_v = \cos\left(\frac{\hat{\gamma}}{2}\right) \mathbf{v}^T q_v - q_0 \sin\left(\frac{\hat{\gamma}}{2}\right) = 0$$

Thus $q_\delta(\mathbf{u}, \mathbf{v}, \mathbf{w})$ is C^2 on $\mathcal{N}$ and its value and those of its derivatives match $q_{\min}(\mathbf{v}, \mathbf{w})$ and its derivatives on $\mathcal{N}$ implying that $q_\delta(\mathbf{u}, \mathbf{v}, \mathbf{w}) = q_{\min}(\mathbf{v}, \mathbf{w})$ on the closure of $\mathcal{N}$, which includes $\mathbf{w} = \mathbf{v}$ and $\mathbf{w}^T \mathbf{v} = -1 + \delta$. □

With Proposition 3.2, the desired quaternion q_d can be found even when $\hat{u}_d = [0 \;\; 0 \;\; -1]^T$, which can be given as:

$$q_d = q_\delta(\hat{r}_d, e_z, \hat{u}_d) \tag{3.38}$$

where $\hat{r}_d$ represents an arbitrary axis of rotation that satisfies $e_z^T \hat{r}_d = 0$ or $\hat{u}_d^T \hat{r}_d$. The resulting quaternion will return the minimal, $\hat{\gamma}$ free, quaternion that rotates e_z to $\hat{u}_d$ in a singular-free and smooth manner. The smoothness of the rotation ultimately depends on

the chosen $\hat{r}_d$; if the time-varying vector $\hat{r}_d(t)$ is as close to the actual axis of rotation from e_z to $\hat{u}_d$ right before and after singularity occurs, the resulting q_d trajectory will be relatively smoother. Similarly, the rare singularity which occurs when u is opposite and parallel to u_d can be avoided with Proposition 3.2 with:

$$\tilde{q}_{\hat{\phi}} = q_\delta(\hat{\tilde{r}}, e_z, R_q^T \frac{u_d}{\|u_d\|}).$$

As before, the smoothness of the rotation depends on the chosen $\hat{\tilde{r}}$. The singularity which occurs when u and u_d are parallel and opposite is a true singularity as there are an infinite amount of rotation axes that can align u with u_d with a rotation angle of $\pm\pi$ in this case. The expression of finding $\tilde{q}$ is merely choosing an arbitrary rotation axis $\hat{\tilde{r}}$ so that the resulting rotation when singular is $\tilde{q} = [0 \;\; \hat{\tilde{r}}^T]^T$.

3.4.3 $\hat{\gamma}$ Singularity

In light of Propositions 3.1 and 3.2, any other quaternion q that rotates unit vector $\mathbf{v}$ to $\mathbf{w}$ can be decomposed to $q_\delta(\mathbf{u}, \mathbf{v}, \mathbf{w})q_{\hat{\gamma}}$ where $q_{\hat{\gamma}} = [\cos(\frac{\hat{\gamma}}{2}) \;\; \sin(\frac{\hat{\gamma}}{2})\mathbf{v}^T]^T$ represents the rotation of $\hat{\gamma}$ about $\mathbf{v}$. This follows from the fact that $qq_\delta(\mathbf{u}, \mathbf{v}, \mathbf{w})^{-1}$ rotates $\mathbf{v}$ onto itself and must necessarily be equal to $q_{\hat{\gamma}}$ for some $\hat{\gamma} \in [-\pi, \pi]$. The following Corollary to Proposition 3.2 can be used to extract the $\hat{\gamma}$ angle from a nonminimal quaternion q by expressing q_δ as $q_\delta(\mathbf{u}, \mathbf{v}, \mathbf{w}) = [c_\delta \;\; s_\delta \hat{r}_\delta^T]^T$, where s_δ and c_δ are each the sine and cosine values associated with the half angle of rotation of q_δ assuming an axis of rotation of $\hat{r}_\delta$..

Corollary 3.1. *Let $q = [q_0 \;\; q_v]$ be a unit quaternion rotating $\mathbf{v}$ to $\mathbf{w}$ and $\mathbf{u}$ a unit vector describing an axis of rotation. Expressing $q_\delta(\mathbf{u}, \mathbf{v}, \mathbf{w}) = [c_\delta \;\; s_\delta \hat{r}^T]^T$, for any $0 < \delta \leq 1$, q can be decomposed into $q_\delta(\mathbf{u}, \mathbf{v}, \mathbf{w})q_{\hat{\gamma}}$ with*

$$\sin(\frac{\hat{\gamma}}{2}) = c_\delta \mathbf{v}^T q_v - s_\delta(q_0 \hat{r}_\delta + \hat{r}_\delta \times q_v)$$
$$\cos(\frac{\hat{\gamma}}{2}) = q_0 c_\delta + s_\delta \hat{r}_\delta^T q_v.$$

In particular, at singularity, when $q_0 = \mathbf{v}^T q_v = 0$*, then* $\hat{\gamma} = 0$*.*

Corollary 3.1 presents a more general definition for $\hat{\gamma}$ previously described in Equation 3.28. More importantly, the definition in Equation 3.28 is subject to a *removable* singularity as $\mathrm{atan2}(q_{v,3}, q_0)$ is undefined when both $q_{v,3}$ and q_0 equals 0. According to Equation 2.7, this case would occur when $\theta = \pm\pi$ and $\hat{r}_3 = 0$. Even so, Proposition 3.2 shows that the singularity of $\hat{\gamma}$ can indeed be removed with the new definition described in Corollary 3.1.

3.5 Overall Stability

The following section describes the overall control scheme while dealing with all singularities such that true global stability can be achieved. The section will go through, step by step, of how the controller is ultimately implemented.

Given an at least twice differentiable translation position trajectory x_d, the non-normalized thrust vector can be given with Equation 3.11. Then a continuous and non-singular input control thrust force f_t can be obtained with Equation 3.12. u_d will subsequently go through the normalization process as described in Equation 3.36 and q_d will be computed according to Equations 3.26 and 3.38. The desired angular velocity ω_d associated with q_d (see Equation 2.11) under normal circumstance, i.e., $\|u_d\| \geq \epsilon$, depends on the derivative of $\hat{u}_d = u_d/\|u_d\|$ given by

$$\dot{\hat{u}}_d = \big(I - \frac{u_d u_d^T}{u_d^T u_d}\big)\frac{\dot{u}_d}{\|u_d\|}.$$

The following Theorem, which is an extension of Theorem 3.2, proves the exponential stability of the overall closed loop control system.

Theorem 3.3. *Consider the closed-loop system described by Equations* 3.7*,* 3.8*,* 3.13 *and* 3.15 *resulting from the control laws defined in Equation 3.6 and Equation 3.11 with* f_t *given with Equation 3.12 and* q_d *generated from* u_d *according to Equation 3.38. Suppose*

that the desired feedforward thrust vector $u_f(t) := \dot{v}_d(t) + ge_z$ *satisfies*

$$\inf_{t\in\mathbb{R}^+} \min \|u_f(t)\| > 0.$$

Then, the overall error system is globally exponentially stable for a sufficiently small ϵ *and large enough* $\lambda_u k_q$*. In particular,* $\tilde{x}$*,* $\tilde{v}$*, and* $\tilde{\omega}$ *converge to zero exponentially starting from any set of initial conditions.*

Proof. Consider the same Lyapunov function candidate from Theorem 3.2:

$$V(\tilde{q}, \tilde{\omega}_r, e_2) = \frac{1}{2}\tilde{\omega}_r^T I_f \tilde{\omega}_r + k_q(\tilde{q}-1)^T(\tilde{q}-1) + e_2^T P e_2. \tag{3.39}$$

In view of Equation 3.1 and Equation 3.24, the time derivative of the presented Lyapunov function is then given by:

$$\begin{aligned}\dot{V}(\tilde{q}, \tilde{\omega}_r, e_2) = &- \tilde{\omega}_r^T K_\omega \tilde{\omega}_r - \tilde{q}^T \Lambda_q^T K_q \tilde{q} - e_2^T Q e_2 \\ &+ 2e_2^T C(\tilde{u}_d + \tilde{u})\end{aligned} \tag{3.40}$$

where u and u_d are each given by,

$$\begin{aligned}\tilde{u} &= (R_q - R_{q_d})e_z \\ \tilde{u}_d &= \|u_d\| R_{q_d} e_z - u_d\end{aligned} \tag{3.41}$$

where the norm of $\tilde{u}_d$ is bounded by 2ϵ. Taking Equation 3.22 into account, the time derivative of V can be also given by

$$\begin{aligned}\dot{V}(\tilde{q}, \tilde{\omega}_r, e_2) = &- \tilde{\omega}_r^T K_\omega \tilde{\omega}_r - (\lambda_q + \lambda_u \|u_d\|^2) k_q \|\tilde{q}_v\|^2 \\ &- e_2^T Q e_2 + 2e_2^T C(\tilde{u}_d + \tilde{u})\end{aligned} \tag{3.42}$$

Invoking Lemma A.2 in Appendix A and $\|\tilde{u}_d\| \leq p2\epsilon$, $\dot{V}(\tilde{q}, \tilde{\omega}_r, e_2)$ can be bounded as

follows:

$$\begin{aligned}\dot{V} \leq & -\alpha\|\tilde{\omega}_r\|^2 - (\lambda_q + \lambda_u \|u_d\|^2)k_q \|\tilde{q}_v\|^2 - \lambda_{\min}(Q) \|e_2\|^2 \\ & + 4c \|e_2\| \|u_d\| \|\tilde{q}_v\| + 2c\|\tilde{u}_d\|\|e_2\| \\ \leq & -\alpha\|\tilde{\omega}_r\|^2 - \lambda_q k_q \|\tilde{q}_v\|^2 + \|\tilde{u}_d\|^2 \\ & - \begin{bmatrix} \|e_2\| & \|u_d\| \|\tilde{q}_v\| \end{bmatrix} \begin{bmatrix} \eta & -2c \\ -2c & \lambda_u k_q \end{bmatrix} \begin{bmatrix} \|e_2\| \\ \|u_d\| \|\tilde{q}_v\| \end{bmatrix}\end{aligned} \tag{3.43}$$

where $\eta = \lambda_{\min}(Q) - c^2$ and $c = \|C\|$. Similar to the proof of Theorem 3.2, if $\eta\lambda_u k_q > 4c^2$, $\dot{V} + k_3 V \leq \|\tilde{u}_d\|^2$ for some $k_3 > 0$. Multiplying both sides of the equation by $e^{k_3\tau}$ and integrating the result from $\tau = 0$ to $\tau = t$ yields $V(t) \leq e^{-k_3 t}V(0) + \frac{4\epsilon^2}{k_3}$. Next we show that the norm of $u_d = \dot{v}_d + ge_z - Ke_2$, given by Equation 3.11 stay above ϵ for a sufficiently large t. By the hypothesis

$$\inf_{t\in\mathbb{R}^+} \|u_f(t)\| = \delta_f$$

for some $\delta_f > 0$ where $u_f(t) = \dot{v}_d(t) + ge_z$ and $\hat{u}_f = \frac{u_f}{\|u_f\|}$. If $\|e_2\| \leq \frac{\delta_f}{2\|K\|}$, it follows that

$$\|u_d\| = \|u_f(t) - Ke_2\| \geq \|u_f(t)\| - \|K\|\|e_2\| \geq \frac{\delta_f}{2}$$

Thus $\|u_d\| \geq \epsilon$ provided that $\epsilon \leq \frac{\delta_f}{2}$. Let t^* be such that $V(0)e^{-k_3 t^*} \leq \frac{\epsilon^2}{k_3}$. Then $V(t) \leq \frac{5\epsilon^2}{k_3}$ implying that $\|e_2(t)\| \leq \frac{\epsilon\sqrt{5}}{\sqrt{k_3\lambda_{\min}(P)}}$, $\forall t \geq t^*$. Choosing

$$\epsilon \leq \frac{\delta_f}{2} \min\{1, \frac{\sqrt{k_3\lambda_{\min}(P)}}{\|K\|\sqrt{5}}\}$$

guarantees that $\|e_2(t)\| \leq \frac{\delta_f}{2\|K\|}$ and consequently $\|u_d(t)\| \geq \epsilon$, $\forall t \geq t^*$. But $\|u_d(t)\| \geq \epsilon \Rightarrow \tilde{u}_d(t) = 0$ making $\dot{V} \leq -k_3 V$ for $t \geq t^*$ proving that $V(t) \leq e^{-k_3(t-t^*)}V(t^*)$, $V(t^*) \leq \frac{5\epsilon^2}{k_3}$. Consequently, V and all the errors converge to zero exponentially. Furthermore, the boundedness of e, v_d, $\dot{v}_d$, and $\ddot{v}_d$ imply that $\dot{q}_d$ and $\bar{\omega}_d = 2\dot{q}_q q_d$ are bounded as well. Therefore, both the linear velocity $v = \dot{x}$ and the angular velocity ω are also bounded. □

3.6 Discussion

3.6.1 Discussion of Λ_q

Λ_q as defined in Equation 3.22 is an additional gain added to guarantee global stability of the overall position control law. Λ_q, when used during control, essentially acts as a second gain for the proportional feedback term $\tilde{q}$. The overall additional proportional gain of the attitude control law due to Λ_q is given by $K_\omega \Lambda_q$, and it varies according to $\|u_d\|$, or desired acceleration of the aircraft. If the desired acceleration is zero in all linear directions (i.e., $\|u_d\| = g$), the amount of additional proportional control is smaller. As desired acceleration increases, the gain increases accordingly.

3.6.2 Usage of $\hat{\gamma}$ and q_{min}

The controller uses the rotation around the global e_z axis, $\hat{\gamma}$, instead of the Euler yaw, γ, to control the yaw torque of the quadrotor. The need to decompose a quaternion into $q_{\hat{\gamma}}$ and $q_{\hat{\phi}}$, instead of simply using the traditional Euler yaw, may be unclear to some; especially given the fact that using $\hat{\gamma}$ introduces yaw, or rotation around the body-fixed $e_{a,z}$ axis. However, the usage of $\hat{\gamma}$ is much needed for the general purpose of the controller. First of all, the main reason the controller is based on quaternions is due to the fact that Euler angles introduce jump singularities. Reverting back to using γ introduces those singularities inherent to Euler angles which makes it undesirable for global purposes. Furthermore, the decomposition of q into $q_{\hat{\gamma}}$ and $q_{\hat{\phi}}$ allows the thrust vector u of the quadrotor to be controlled independent from the yaw of the quadrotor as $q_{\hat{\phi}}$ directly represents the minimal quaternion rotation from e_z to u (i.e., $q_{min}(e_z, u)$). Accurate control of u is much more important than maintaining the Euler yaw constant as the control scheme is fundamentally designed to control the linear position of the quadrotor.

3.6.3 Discussion of $\tilde{q}$

In this chapter, 2 ways of finding $\tilde{q}$ was introduced. The first method involves finding $q_{\hat{\phi},d}$ with Equation 3.26, and the $\hat{\gamma}_d$ information is subsequently added with Equation 3.29. Finally, the attitude error $\tilde{q}$ can be found with Equation 3.1. The second method involves finding $\tilde{q}_{\hat{\phi}}$ directly from u_d with Equation 3.33, and subsequently adding $\tilde{q}_{\hat{\gamma}}$ with Equation 3.34 after extracting $\hat{\gamma}$ from q with Equation 3.28. Both methods are ultimately obtained as follows:

$$\begin{aligned} \tilde{q}_{[1]} &= \left(q_{\hat{\phi}_d} q_{\hat{\gamma}_d}\right)^{-1} q \\ \tilde{q}_{[2]} &= \tilde{q}_{\hat{\gamma}} \tilde{q}_{\hat{\phi}} \end{aligned} \tag{3.44}$$

where $\tilde{q}_{[1]}$ and $\tilde{q}_{[2]}$ each represent $\tilde{q}$ found from methods 1 and 2, and $\tilde{q}_{\hat{\gamma}} = q_{\hat{\gamma}_d}^{-1} q_{\hat{\gamma}}$. Note that the resulting q_d from $\tilde{q}_{[1]}$ and $\tilde{q}_{[2]}$ are not equivalent (i.e., $q\tilde{q}_{[1]}^{-1} \neq q\tilde{q}_{[2]}^{-1}$); in other words, the transient response may be different with the exact same trajectory. Even so, the q, and q_d, at steady-state (i.e., $\tilde{q} = 0$) would not differ as both attitude errors aims to rotate q to align u with u_d at a fixed $\hat{\gamma}_d$. The following proposition explains the difference between the two attitude errors:

Proposition 3.3. *The difference between the two attitude errors, $\tilde{q}_{[2]}$ and $\tilde{q}_{[1]}$, can be expressed as a pure rotation around the e_z axis:*

$$\tilde{q}_{[2]} \tilde{q}_{[1]}^{-1} = q_{\bar{\gamma}} = \begin{bmatrix} \cos\frac{\bar{\gamma}}{2} \\ 0 \\ 0 \\ \sin\frac{\bar{\gamma}}{2} \end{bmatrix}$$

where $\bar{\gamma}$ is given by,

$$\bar{\gamma} = 2\,\mathrm{atan2}\,(\frac{u_{d,1}u_2 - u_{d,2}u_1}{\sigma_{u_d^T u}\sigma_u}, \frac{1 + u_3 + u_{d,3} + u_d^T u}{\sigma_{u_d^T u}\sigma_u}).$$

Proof. Let $q_{d_{[1]}}$ and $q_{d_{[2]}}$ each be the resulting desired quaternion attitude when the quater-

nion error are each $\tilde{q}_{[1]}$ and $\tilde{q}_{[2]}$, each defined as in Equation 3.44, the both $q_{d_{[1]}}$ and $q_{d_{[2]}}$ can be given by:

$$q_{d_{[1]}} = q\left(\left(q_{\hat{\phi}_d} q_{\hat{\gamma}_d}\right)^{-1} q\right)^{-1} = q_{\hat{\phi}_d} q_{\hat{\gamma}_d}$$
$$q_{d_{[2]}} = q\left(\tilde{q}_{\hat{\gamma}} \tilde{q}_{\hat{\phi}}\right)^{-1} = q\tilde{q}_{\hat{\phi}}^{-1} \tilde{q}_{\hat{\gamma}}^{-1}$$

As mentioned before $\tilde{q}_{\hat{\phi}}$ and $q_{\hat{\phi}_d}$ each represent the minimal $\hat{\gamma}$ free rotation of $Rq^T u_d \rightarrow e_z$ and $e_z \rightarrow u_d$, respectively. With $\tilde{q}_{\hat{\phi}}$, a quaternion that represents a consecutive rotation from $e_z \rightarrow u$ then $u \rightarrow u_d$ can be described as follows:

$$\begin{aligned} q_{Rq^T u_d \rightarrow e_z} &= \tilde{q}_{\hat{\phi}} \\ q_{u_d \rightarrow u} &= q\tilde{q}_{\hat{\phi}} q^{-1} \\ q_{e_z \rightarrow u \rightarrow u_d} &= q_{u_d \rightarrow u}^{-1} q_{e_z \rightarrow u} = \left(q\tilde{q}_{\hat{\phi}} q^{-1}\right)^{-1} q_{\hat{\phi}} = q\tilde{q}_{\hat{\phi}}^{-1} q^{-1} q_{\hat{\phi}}. \end{aligned}$$

where $q_{\hat{\phi}}$ denotes the minimal $\hat{\gamma}$ free rotation of $e_z \rightarrow u$. Although both quaternions represent a rotation from e_z to u_d, $q_{\hat{\phi}_d}$, as stated above, is $\hat{\gamma}$ free, while $q_{e_z \rightarrow u \rightarrow u_d}$ is not. If $\hat{\gamma}$ is removed, according to Proposition 3.1, both quaternions would represent the same rotation. Hence,

$$q_{\hat{\phi}_d} = q_{e_z \rightarrow u \rightarrow u_d} q_{\hat{\gamma}_{e_z \rightarrow u \rightarrow u_d}}^{-1} \tag{3.45}$$

where $q_{\hat{\gamma}_{e_z \rightarrow u \rightarrow u_d}}$ represents a pure rotation around the e_z axis extracted from $q_{e_z \rightarrow u \rightarrow u_d}$ according to Proposition 3.1.

The difference between $\tilde{q}_{[1]}$ and $\tilde{q}_{[2]}$, or $q_{d_{[1]}}$ and $q_{d_{[2]}}$, can be obtained as follows:

$$\begin{aligned} \tilde{q}_{[2]} \tilde{q}_{[1]}^{-1} = q_{d_{[2]}}^{-1} q_{d_{[1]}} &= \left(q\tilde{q}_{\hat{\phi}}^{-1} \tilde{q}_{\hat{\gamma}}^{-1}\right)^{-1} q_{\hat{\phi}_d} q_{\hat{\gamma}_d} \\ &= \tilde{q}_{\hat{\gamma}} \tilde{q}_{\hat{\phi}} q^{-1} q_{\hat{\phi}_d} q_{\hat{\gamma}_d}. \end{aligned} \tag{3.46}$$

By inserting $q_{\hat{\phi}_d}$ from Equation 3.45 into Equation 3.46, the following result can be ob-

tained:

$$\begin{aligned} q_{d_{[2]}}^{-1} q_{d_{[1]}} &= \tilde{q}_{\hat{\gamma}} \tilde{q}_{\hat{\phi}} q^{-1} q_{e_z \to u \to u_d} q_{\hat{\gamma}_{e_z \to u \to u_d}}^{-1} q_{\hat{\gamma}_d} \\ &= \tilde{q}_{\hat{\gamma}} \tilde{q}_{\hat{\phi}} q^{-1} q \tilde{q}_{\hat{\phi}}^{-1} q^{-1} q_{\hat{\phi}} q_{\hat{\gamma}_{e_z \to u \to u_d}}^{-1} q_{\hat{\gamma}_d} \\ &= q_{\hat{\gamma}_d}^{-1} q_{\hat{\gamma}_{e_z \to u \to u_d}}^{-1} q_{\hat{\gamma}_d}. \end{aligned}$$

Since the three consecutive rotations of $q_{d_{[2]}}^{-1} q_{d_{[1]}}$ are all around the same rotation axis, the e_z axis and also the first and third rotation angle are the same, $\hat{\gamma}_d$, the difference between $q_{d_{[1]}}$ and $q_{d_{[2]}}$ can be further simplified as:

$$q_{d_{[2]}}^{-1} q_{d_{[1]}} = q_{\hat{\gamma}_{e_z \to u \to u_d}}^{-1}.$$

Therefore, the difference between the two attitude errors, $\tilde{q}_{[2]}$ and $\tilde{q}_{[1]}$, is a pure rotation around the e_z axis with an angle of $\bar{\gamma}$, where the angle $\bar{\gamma}$ is given by:

$$\bar{\gamma} = q_{\hat{\gamma}_{e_z \to u \to u_d}}^{-1}.$$

$\bar{\gamma}$ can now be easily obtained in terms of $u = [u_1 \;\; u_2 \;\; u_3]^T$ and $u_d = [u_{d,1} \;\; u_{d,2} \;\; u_{d,3}]^T$. Recall that $q_{\hat{\gamma}_{e_z \to u \to u_d}} = q_{u_d \to u}^{-1} q_{e_z \to u}$, the two minimal rotations $q_{u_d \to u}$ and $q_{e_z \to u}$ can be expressed in terms of u and u_d, according to Equation 2.13, as:

$$q_{e_z \to u} = \frac{1}{\sigma_u} \begin{bmatrix} 1 + u_3 \\ -u_2 \\ u_1 \\ 0 \end{bmatrix} \quad q_{u_d \to u} = \frac{1}{\sigma_{u_d^T u}} \begin{bmatrix} 1 + u_d^T u \\ u_{d,2} u_3 - u_{d,3} u_2 \\ u_{d,3} u_1 - u_{d,1} u_3 \\ u_{d,1} u_2 - u_{d,2} u_1 \end{bmatrix}$$

where $\sigma_u = \sqrt{2(1 + u_3)}$ and $\sigma_u = \sqrt{2(1 + u_d^T u)}$. With the two minimal quaternions, $q_{\hat{\gamma}_{e_z \to u \to u_d}}$ can now be obtained as follows:

$$
\begin{aligned}
q_{e_z \to u \to u_d} &= \frac{1}{\sigma_{u_d^T u}} \begin{bmatrix} 1 + u_d^T u \\ u_{d,2}u_3 - u_{d,3}u_2 \\ u_{d,3}u_1 - u_{d,1}u_3 \\ u_{d,1}u_2 - u_{d,2}u_1 \end{bmatrix}^{-1} \frac{1}{\sigma_u} \begin{bmatrix} 1 + u_3 \\ -u_2 \\ u_1 \\ 0 \end{bmatrix} \\
&= \frac{1}{\sigma_{u_d^T u}\sigma_u} \begin{bmatrix} 1 + u_d^T u \\ u_{d,2}u_3 - u_{d,3}u_2 \\ u_{d,3}u_1 - u_{d,1}u_3 \\ u_{d,1}u_2 - u_{d,2}u_1 \end{bmatrix}^{-1} \begin{bmatrix} 1 + u_3 \\ -u_2 \\ u_1 \\ 0 \end{bmatrix}
\end{aligned}
$$

The proof can be completed by showing that $\bar{\gamma} = q^{-1}_{\hat{\gamma}_{e_z \to u \to u_d}}$, with $q_{e_z \to u \to u_d}$, is given by:

$$
\begin{aligned}
\bar{\gamma} &= 2\,\mathrm{atan2}\left(-q_{\hat{\gamma}_{e_z \to u \to u_d},v,3}, q_{\hat{\gamma}_{e_z \to u \to u_d},0}\right) \\
&= 2\,\mathrm{atan2}\left(\frac{u_{d,1}u_2 - u_{d,2}u_1}{\sigma_{u_d^T u}\sigma_u}, \frac{1 + u_3 + u_{d,3} + u_d^T u}{\sigma_{u_d^T u}\sigma_u}\right).
\end{aligned}
$$

□

Assuming $1 + u_3 + u_{d,3} + u_d^T u \neq 0$, $\bar{\gamma}$ is relatively small if $u_{d,1}u_2 - u_{d,2}u_1$ is kept close to zero. Recall that $u_{d,1}u_2 - u_{d,2}u_1$ is the third vector component of minimal rotation of $u_d \to u$ (i.e., $q_{u_d \to u,v,3}$). Hence, if the z-component of the rotation axis of $u_d \to u$ is kept close to zero, the difference between $\tilde{q}_{[1]}$ and $\tilde{q}_{[2]}$, or $q_{d_{[1]}}$ and $q_{d_{[2]}}$, is negligible.

3.6.4 Differential Flatness

The designed controller, much like other controllers, uses the differential flatness property of the quadrotor. The linkage between the outer position loop and the inner attitude loop is based on the premise that both the translational position states, and its derivatives, and the attitude states, and its derivatives, can be expressed in terms of the thrust vector u_d, or the normalized $\hat{u}_d$, and the yaw angle, as defined in Proposition 3.1, and the derivatives of the two variables. The two control laws directly reflect such differential flatness. In case of the position control law defined in Equation 3.11, the input and translation state variables

of the quadrotor are related to the desired thrust vector u_d without the introduction of any integration of the thrust vector. Subsequent conversion to attitude configurations also only requires the thrust vector and its derivatives. The attitude control law as defined in Equation 3.6, on the other hand, relates the attitude errors to the three torque input variables without integration of the attitude, which relates to the thrust vector. In fact, the differential flatness property of the quadrotor allows accurate control of the translational position and yaw, or the four flat outputs, with only a PD attitude controller. The advantage of utilizing the different methods discussed in previous sections is in that such approximation and smoothing allows the avoidance of all singularities that may occur during state transformations based on differential flatness of quadrotors. Many controllers have attempted to solve such singularities; however, few have demonstrated aggressive flight trajectory tracking which involves 360° flipping maneuvers which passes through the singularity that occurs when the attitude is completely flipped [51]. The discussed controller solve such singularity in theory and in practice by demonstrating similar maneuvers in future sections.

CHAPTER 4

SIMULATIONS

The subsequent simulation of the proposed controller was done based on velocity control, instead of position control. This decision was primarily made to allow the quadrotor to be used with a Radio-Controlled (RC) transmitter/receiver, as all implementation were planned to be done in an outdoor environment. The velocity control law can be deduced from u_d simply by setting $K_i = 0_{3\times3}$:

$$u_d = \dot{v}_d - K_v\tilde{v} - K_x\tilde{x} + ge_z. \tag{4.1}$$

Note that while the position control law was PID control with 3 feedback terms, the new velocity control law is only PI control with 2 feedback terms. The resulting transient response is expected to differ as the PI velocity control law does not include any derivative feedback; however, the stability of both control laws do not change as long as the gains of both control laws are chosen to be stable (i.e., chosen to make Equation 3.19 Hurwitz). A fixed yaw value, $\hat{\gamma}_d$, was chosen and remained constant for the remainder of the flight; and the x and y directions of the reference velocities were aligned with the fixed yaw. The controller was designed so that the reference inputs would control the linear velocity of the aircraft so that the aircraft north would align with the reference x-direction, and west with the reference y-direction. This can be accomplished by rotating the global frame by the fixed yaw in the following manner:

$$v_1 = v_1^{\mathcal{I}} \cos\hat{\gamma}_d + v_2^{\mathcal{I}} \sin\hat{\gamma}_d$$
$$v_2 = v_1^{\mathcal{I}} \sin\hat{\gamma}_d - v_2^{\mathcal{I}} \cos\hat{\gamma}_d.$$

This way giving a positive reference velocity v_d would ensure that the quadrotor moves in the positive direction relative to its body-fixed frame, or the yaw-fixed frame.

Figure 4.1 shows a detailed schematic of the implemented final controller. The controller initially receives desired velocity values and passes them onto the outer velocity and the inner attitude controllers. Both the torque control input τ_a and the desired thrust $f_t = m\|u_d\|$ are eventually obtained with the two control laws; and, with Equation 2.21, the rotor velocities Ω are subsequently obtained and fed into the quadrotor plant. As shown in Figure 4.1, the attitude error quaternion $\tilde{q}$ was obtained with Equation 3.33, instead of Equation 3.26. The angle $\bar{\gamma}$ stays close to zero for most velocity trajectories; therefore, the desired quaternion q_d obtained from both methods are approximately the same. The advantage of using Equation 3.33 is in that the method eliminates the need to consider the singularity that occurs when $u_d = [0 \;\; 0 \;\; -1]^T$ as explained in Section 3.3. $\tilde{\omega}$ was obtained accordingly with

$$\dot{\hat{u}}_d = \frac{1}{\|u_d\|}(1 - \hat{u}_d\hat{u}_d^T)(R_q^T\dot{u}_d - \omega_b \times u_d).$$

If the assumption that $\bar{\gamma} \approx 0$ is held true then the z-component of $\tilde{\omega}$ should track the z-component of ω closely (i.e., $\omega_{d,3} \approx 0$) as the resulting $\hat{\gamma}_d$ from $\tilde{q}$ should be fixed as long as $\bar{\gamma}$ is not introduced.

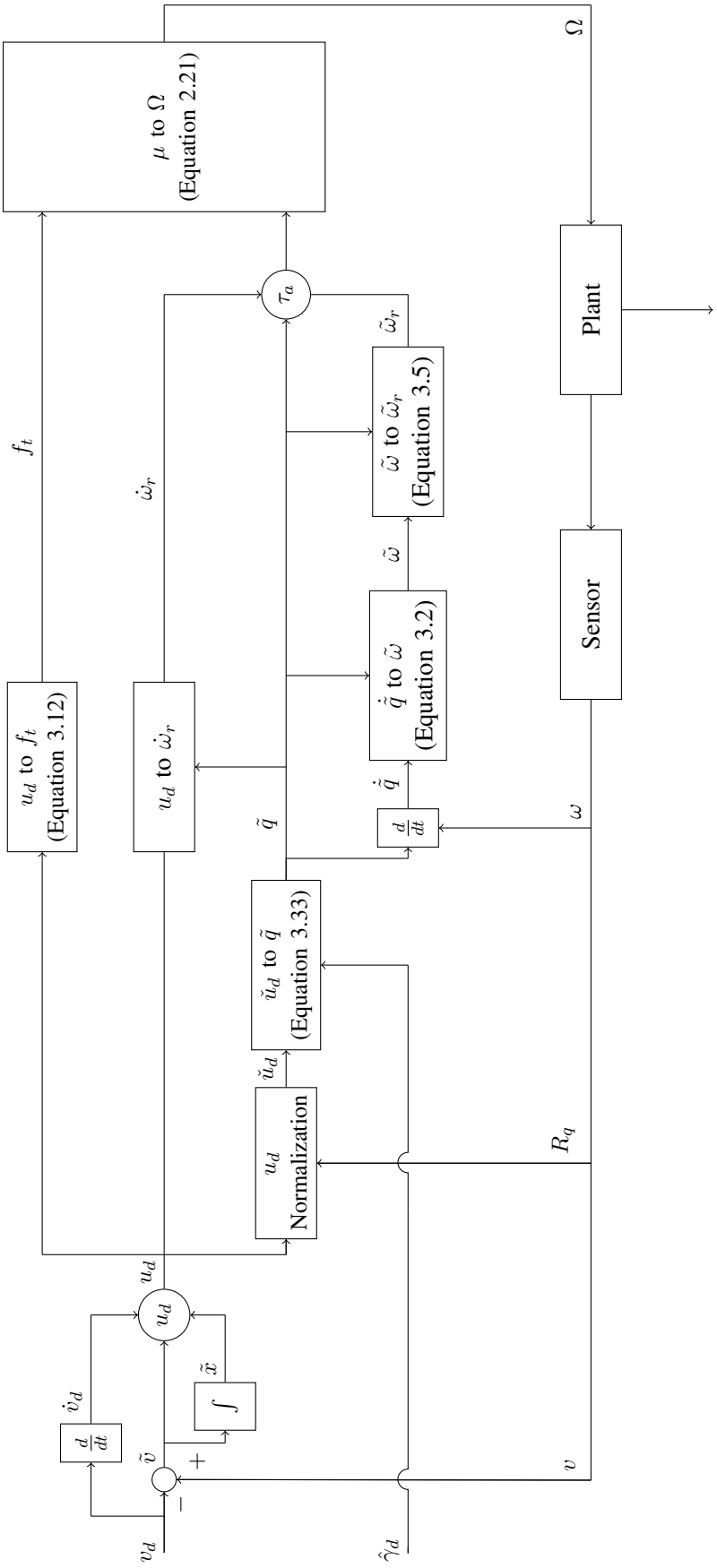


Figure 4.1: Schematic of Final Controller

4.1 Simulation Details

This section will briefly go over some important simulation, and implementation, details, that completes the description of the controller used to obtain the results in future sections. The section will discuss various filters and approximations implemented along with the outlined control scheme shown in Figure 4.1.

4.1.1 Implemented Filters

Minimal amount of filters were applied as the sensor values received from the Extended Kalman Filter (EKF) had minimal noise. Filters, for the most part, were only implemented when the derivatives, especially derivatives of sensor values, were required. Most derivatives that were not directly available were either numerically or analytically differentiated. Analytical differentiation does not require any filtering if all values involved in the differentiation are smooth and if the values do not introduce to much noise. Most derivatives necessary to the control scheme were analytically derived with the exception of $\dot{\omega}_r$, which was obtained by passing ω_r through a 20Hz low pass derivative filter.

The raw reference inputs are not guaranteed to be smooth, a great example of this would be a step input which jumps from one value to another within one sample time. To this end, a third order filter was applied to the raw v_d input to ensure an at least twice continuously differentiable v_d; both $\dot{v}_d$ and $\ddot{v}_d$ used in the controller corresponded to the filtered v_d instead of the raw v_d input. The design of the third order filter used can be found in Appendix B. The specific values for each of the filter parameters are $T = 0.05$, $\zeta = 0.8$ and $\omega_n = \frac{10}{3}$. Figure 4.2 shows the filtered result of a step input with an amplitude of 1. The filtered trajectory shown in red, though slower than the unfiltered trajectory shown in black, is much more realistic to track and ensures that the two required derivatives $\dot{v}_d$ and $\ddot{v}_d$ are also smooth and continuous.

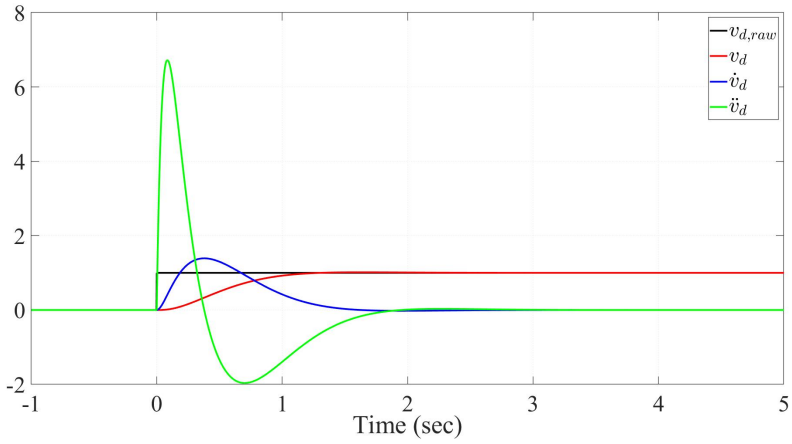


Figure 4.2: Filtering of v_d

4.1.2 $\hat{\gamma}$ Approximation

As mentioned before, the attitude error, $\tilde{q}$, was obtained with Equation 3.33, instead of Equation 3.26. This method requires the extraction of $\hat{\gamma}$ from the actual quaternion q as it is needed when computing the overall attitude error with Equation 3.34. However, the method of extracting the angle $\hat{\gamma}$ from any quaternion with Equation 3.28 is singular when $\hat{\theta} = \pm\pi$ and $\hat{r}_3 = 0$ according to Equation 2.7. According to Proposition 3.2, the singular point of $\hat{\gamma}$ is a removable singularity and the angle can be extracted without singularity with Corollary 3.1. The smoothness of Corollary 3.1 heavily relies on the orthogonal unit vector p chosen. To this end, let us decompose an arbitrary quaternion q into:

$$q = q_\delta(\check{r}, e_z, u)q_{\check{\gamma}} \tag{4.2}$$

where $\check{\gamma}$ is a close approximation of the actual $\hat{\gamma}$, and $\check{r}$ is the resulting axis of rotation from the rotation $qq_{\check{\gamma}}^{-1}$. In this case, $q_{\hat{\gamma}} = q_\delta(\check{r}, e_z, u)^{-1}q$, and if $\check{r}$ happens to be the common perpendicular of e_z and u, then $q_\delta(\check{r}, e_z, u) = q_{\min}(e_z, u)$ even in the singular case. According to Corollary 3.1, $\hat{\gamma}$ found according to the decomposition described in

Equation 4.2 can be obtained with:

$$\begin{aligned}\cos\left(\frac{\hat{\gamma}}{2}\right) &= \cos\left(\frac{\hat{\phi}}{2}\right) q_0 + \sin^2\left(\frac{\hat{\phi}}{2}\right) \cos\left(\frac{\check{\gamma}}{2}\right) \\ \sin\left(\frac{\hat{\gamma}}{2}\right) &= \cos\left(\frac{\hat{\phi}}{2}\right) e_z^T q_v + \sin^2\left(\frac{\hat{\phi}}{2}\right) \sin\left(\frac{\check{\gamma}}{2}\right)\end{aligned} \tag{4.3}$$

where $\cos\left(\frac{\hat{\phi}}{2}\right) = \sqrt{q_0^2 + (e_z^T q_v)^2}$. Note that with Equation 3.37, Equation 4.3 can be easily shown to result in actual $\hat{\gamma}$ value whenever $\check{\gamma} = \hat{\gamma}$. This expression ultimately solves the singularity issue present in Equation 3.28 as $\sin\left(\frac{\hat{\gamma}}{2}\right) = \sin\left(\frac{\check{\gamma}}{2}\right)$ and $\cos\left(\frac{\hat{\gamma}}{2}\right) = \cos\left(\frac{\check{\gamma}}{2}\right)$ whenever $\hat{r}_3 = 0$ and $\theta = \pm\pi$, or $q_0 = e_z^T q_v = 0$. Also, the smoothness and accuracy of the approximation depends on the value $\check{\gamma}$ chosen. The closest available value for $\check{\gamma}$ is the $\hat{\gamma}$ value from the past; in the case of an discretized controller, the $\hat{\gamma}$ value from one previous time step (i.e., $\hat{\gamma}_{(n-1)}$) can be chosen. Another option would be to use the desired angle $\hat{\gamma}_d$ assuming that the desired angle and actual yaw angles remain the same or small in difference (i.e. $\tilde{\hat{\gamma}} = \hat{\gamma} - \hat{\gamma}_d \approx 0$). This assumption is reasonable as yaw control is only implemented to keep a constant angle; even with disturbances the deviation will be small.

Furthermore, $\hat{\gamma}$ approximation is only needed when using the second method of finding $\tilde{q}$ (i.e., directly finding $\tilde{q}$ with $R_q^T u_d$). A similar approximation is needed when using the first method (i.e., finding $\tilde{q}$ by first building q_d with u_d); however, in the case of the first method, the q_d will be built with approximation instead of $\hat{\gamma}$. Similar to the second method, the approximation can be done with Equation 4.2 and choosing $\check{r}$ as the desired axis of rotation from one previous time step (i.e., $\hat{r}_{d,(n-1)}$). The second method of finding $\tilde{q}$ was chosen primarily because, due to the nature of the controller, it was more important to accurately track the actual u_d, or the minmial q_d, than to keep a constant $\hat{\gamma}_d$. Even though $\hat{\gamma}$ is approximated near singularity, $\tilde{q}_{\hat{\phi}}$ accurately obtains a minimal quaternion that rotates u to u_d. The error added due to $\hat{\gamma}$ approximation and $\bar{\gamma}$ defined in Proposition 3.3 can be seen as small changes made to $\hat{\gamma}_d$.

4.1.3 Simulation Model

Simulations were done based on Software in the Loop (SITL). The simulation model used, as well as the EKF library, was borrowed from a open-source Library (Ardupilot: http://ardupilot.org/ardupilot/). In addition, the conversion from aircraft torques to rotor angular velocity values, or PWM values, were modified from the original Ardupilot library to be consistent with the rotor torque model described in Equations 2.19 and 2.20. Instead of using traditional model parameters (i.e., inertial matrix, drag and lift coefficient, etc.), the simulation software defined three parameters, which are related to the physical parameters of a quadrotor, to illustrate the simulation model. Additional modifications were made to the original SITL model to fully reflect quadrotor dynamics. The modified simulation model assumes a diagonal inertial matrix:

$$I_f = \begin{bmatrix} I_{xx} & 0 & 0 \\ 0 & I_{yy} & 0 \\ 0 & 0 & I_{zz} \end{bmatrix}$$

where $I_{xx} = I_{yy} = \frac{I_{zz}}{2}$. When assuming a diagonal inertial matrix and ignoring the Gyroscopic term G_a, the attitude dynamic equation defined as Equation 2.17 can be simplified as:

$$I_{xx}\dot{\omega}_{b,1} = \tau_{a,1} + (I_{zz} - I_{yy})\dot{\omega}_{b,2}\dot{\omega}_{b,3}$$
$$I_{yy}\dot{\omega}_{b,2} = \tau_{a,2} + (I_{xx} - I_{zz})\dot{\omega}_{b,2}\dot{\omega}_{b,3}$$
$$I_{zz}\dot{\omega}_{b,3} = \tau_{a,3} + (I_{yy} - I_{xx})\dot{\omega}_{b,3}\dot{\omega}_{b,2}.$$

With the assumption $I_{xx} = I_{yy} = \frac{I_{zz}}{2}$, the simulation model can be further simplified as:

$$\dot{\omega}_{b,1} = \frac{\tau_{a,1}}{I_{xx}} + \dot{\omega}_{b,2}\dot{\omega}_{b,3}$$
$$\dot{\omega}_{b,2} = \frac{\tau_{a,2}}{I_{yy}} - \dot{\omega}_{b,2}\dot{\omega}_{b,3}$$
$$\dot{\omega}_{b,3} = \frac{\tau_{a,3}}{I_{zz}}$$

where τ_a is defined according to Equation 2.20. The simplified dynamic equation can be alternatively represented as:

$$\dot{\omega}_{b,1} = \frac{bl}{I_{xx}}\bar{\Omega}_1 + \dot{\omega}_{b,2}\dot{\omega}_{b,3}$$
$$\dot{\omega}_{b,2} = \frac{bl}{I_{yy}}\bar{\Omega}_2 - \dot{\omega}_{b,2}\dot{\omega}_{b,3}$$
$$\dot{\omega}_{b,3} = \frac{\kappa}{I_{zz}}\bar{\Omega}_3$$

where $\bar{\Omega} = [\bar{\Omega}_1 \ \bar{\Omega}_2 \ \bar{\Omega}_3]^T$ denotes a vector that represents the sum, or difference, of the element-squared rotor velocities for each of the corresponding aircraft torques:

$$\bar{\Omega}_1 = \Omega_2^2 - \Omega_4^2$$
$$\bar{\Omega}_2 = \Omega_1^2 - \Omega_3^2$$
$$\bar{\Omega}_3 = \Omega_1^2 - \Omega_2^2 + \Omega_3^2 - \Omega_4^2.$$

The simulation model can be further simplified by using rotor values normalized with the square of the maximum angular velocity speed Ω_{max}. Hence, the final dynamic model of the simulation can be expressed as:

$$\dot{\omega}_{b,1} = \frac{bl\Omega_{max}^2}{I_{xx}}\hat{\bar{\Omega}}_1 + \dot{\omega}_{b,2}\dot{\omega}_{b,3}$$
$$\dot{\omega}_{b,2} = \frac{bl\Omega_{max}^2}{I_{yy}}\hat{\bar{\Omega}}_2 - \dot{\omega}_{b,2}\dot{\omega}_{b,3} \tag{4.4}$$
$$\dot{\omega}_{b,3} = \frac{\kappa\Omega_{max}^2}{I_{zz}}\hat{\bar{\Omega}}_3$$

where $\hat{\bar{\Omega}} = \frac{\bar{\Omega}}{\Omega_{max}^2}$. The translation dynamic model of the simulation model follows the equation defined in Equation 2.15. However, the thrust force f_t, shown in Equation 2.19, must be modified to reflect the normalization of the rotor velocities. To this end, the simulation model introduces the hover throttle ratio, which essentially defines the percentage of the maximum thrust force needed to hover a quadrotor with mass m. The hover throttle

ratio, T_h, can be defined as:

$$T_h = \frac{mg}{4T_{max}} \tag{4.5}$$

where $T_{max} = b\Omega_{max}^2$. Note that the torque and thrust forces were obtained from the individual angular velocities of the four rotors. In reality, such process would involve finding the $\mu = [f_t \ \ \tau_a]^T$ according to Equation 2.21. In the case of the particular simulation model used, the matrix M used in Equation 2.21 must be replaced with:

$$M = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 0 & 1 & 0 & -1 \\ 1 & 0 & -1 & 0 \\ 1 & -1 & 1 & -1 \end{bmatrix}$$

to reflect the normalization of the four rotor velocities. The model parameters, in accordance with the standard model described in Equations 2.14 to 2.17 and 2.21, are listed in Table 4.1; the parameters were determined in conformity with the simplified model described in Equation 4.4 and $\Omega_{max} = 1000$ rad/sec.

Table 4.1: Model Parameters of Simulation Aircraft

Property	**Parameter**	**Value**
Mass in kg	m	1.5
Moment of Inertia about x-axis in $kg \cdot m^2$	I_{xx}	2.1×10^{-2}
Moment of Inertia about y-axis in $kg \cdot m^2$	I_{yy}	2.1×10^{-2}
Moment of Inertia about z-axis in $kg \cdot m^2$	I_{zz}	4.2×10^{-2}
Propeller Lift Coefficient	b	1.226×10^{-5}
Propeller Drag Coefficient	κ	5.864×10^{-7}
Arm Length in m	l	0.3

4.1.4 Simulation Gains

Table 4.2 shows the controller gains used during simulation. While the angular velocity gain K_ω can be any symmetric positive definite matrix, the quaternion gain K_q must have a single gain (i.e., $K_q = [0_{3\times1} \;\; k_q I_{3\times3}]$). Also, note that the two variable gains λ_q and λ_u were chosen to satisfy the condition described in the proof of Theorem 3.2. Finally, due to the reference angular velocity error term ω_r, the quaternion gain K_q does not reflect the final gain for the attitude state error $\tilde{q}$; in fact, the attitude control law, in terms of $\tilde{q}$ and $\tilde{\omega}$, can be alternatively given as:

$$\tau_a = I_f\dot{\omega}_d - I_f(\Lambda_q\dot{\tilde{q}}) + \omega_d \times I_f\omega_b - (\Lambda_q\tilde{q}) \times I_f\omega_b + G_a - K_\omega\tilde{\omega} - (K_q + K_\omega\Lambda_q)\tilde{q}$$

where the actual gain of $\tilde{q}$ is $(K_q + K_\omega\Lambda_q)$. Even though the original K_q gain does not have much flexibility as all three vector components of the quaternion error $\tilde{q}$ has a single gain, such flexibility can be introduced to the attitude error with the addition of $K_\omega\Lambda_q$.

Table 4.2: Controller Gains, Simulation

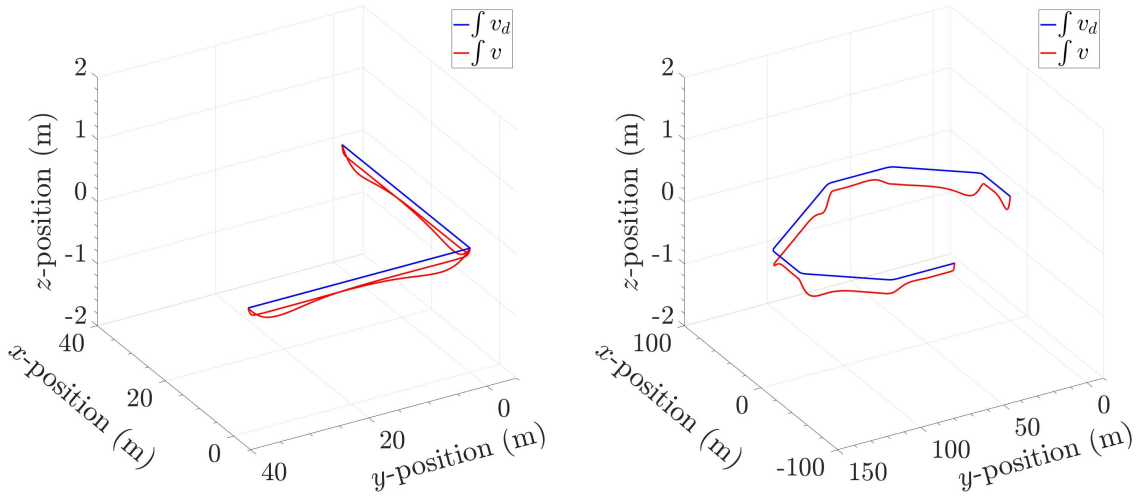
Gain Description	**Parameter**	**Value**
Linear Velocity	K_v	$2.5I_{3\times3}$
Linear Position	K_x	$4I_{3\times3}$
Quaternion	K_q	$[0_{3\times1} \;\; 2.3333I_{3\times3}]$
ω_r	K_ω	$0.3333I_{3\times3}$
Variable Gain 1	λ_q	0.0208
Variable Gain 2	λ_u	0.0521

4.2 Simulation Results

The following simulation results will be structured in the order of simple to complex trajectories. Note that all simulation testing were conducted while controlling $\hat{\gamma}$ at 0° (i.e., $\hat{\gamma}_d = 0$).

4.2.1 Step Response

The subsequent results show the performance of the controller under two different step trajectories. The first trajectory involves simple uni-directional step velocity inputs which moves the quadrotor forward, backwards, and sideways. The second trajectory consists of eight consecutive steps where the reference velocities change both in the x and y direction simultaneously. Figure 4.3 displays a visual representation of the two step trajectories. Note that the trajectory plots, which describes the quadrotor translation movement in terms of the integral of velocity, does not imply that the position of the quadrotor was actually controlled. The trajectory plots were only included to better illustrate the positional movement of the quadrotor while the quadrotor tracks the velocity trajectory v_d.



(a) Trajectory 1 (b) Trajectory 2

Figure 4.3: Test Step Trajectories

4.2.1.1 Step Trajectory 1

Figures 4.4 and 4.6 displays the step velocity response of the quadrotor when tracking step trajectory 1; each figure illustrates the quadrotor tracking step velocity inputs in the x and y directions. The corresponding attitude and angular velocity responses are shown in Figures 4.5 and 4.7. Step Trajectory 1, for both x and y directions, accelerates the quadrotor to 5 m/s from rest, returns to 0 m/s, decelerates to -5 m/s, and returns to rest. Additionally, as evident in Figures 4.4 and 4.6, the desired z velocity, $v_{d,3}$ were fixed at zero throughout the whole trajectory. Recall that raw reference inputs were filtered with a third order low pass filter; the effects of filtering can be seen in v_d of the step trajectories as the velocity trajectories shown in Figures 4.4 and 4.6 are displayed as filtered step inputs instead of a jump from one velocity to another.

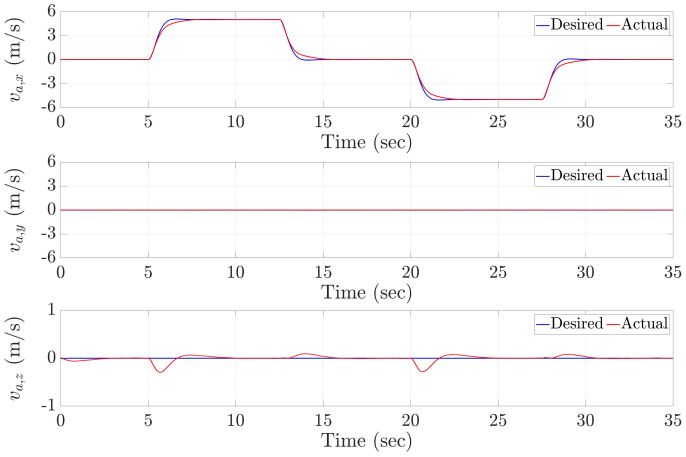
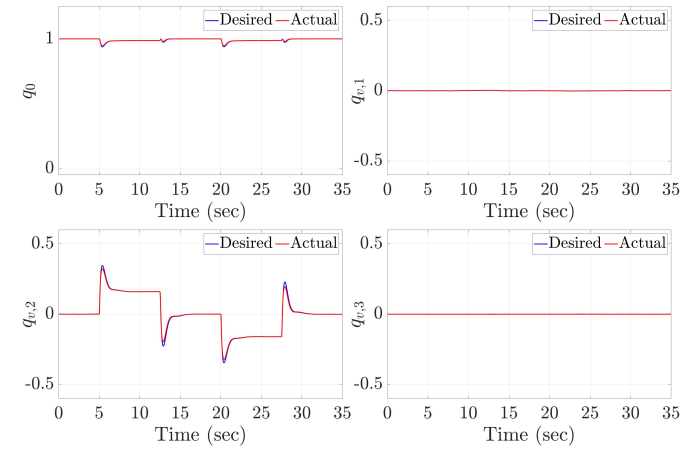
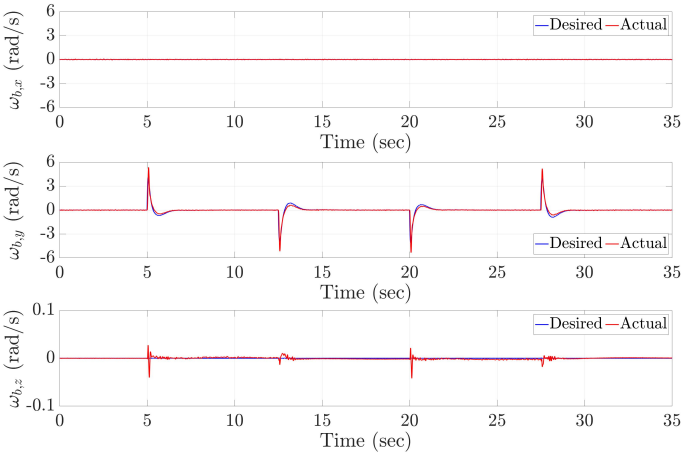


Figure 4.4: Velocity Response to Step Trajectory in the x-Direction



(a) Quaternion Attitude



(b) Angular Velocity

Figure 4.5: Attitude Response of Step Trajectory 1 in the x-Direction

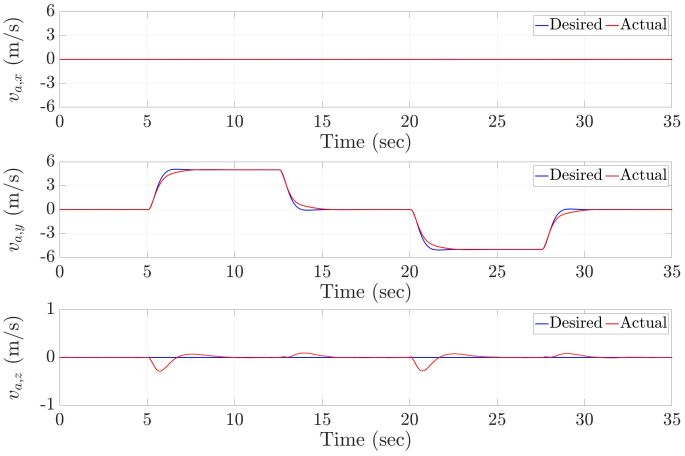
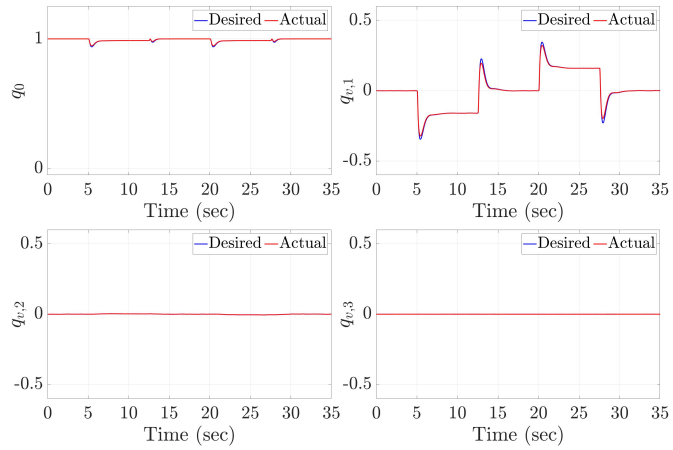
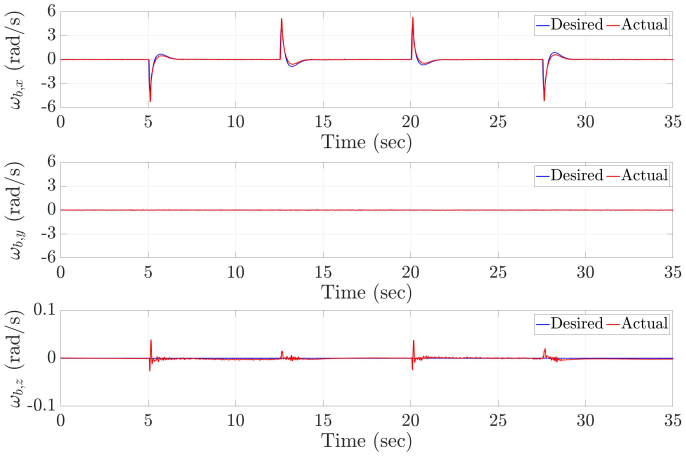


Figure 4.6: Velocity Response of Step Trajectory 1 in the y-Direction



(a) Quaternion Attitude



(b) Angular Velocity

Figure 4.7: Attitude Response of Step Trajectory 1 in the y-Direction

4.2.1.2 Step Trajectory 2

Figure 4.8 displays the velocity response of the quadrotor when tracking step trajectory 2, shown in Figure 4.3; and the corresponding attitude and angular velocity responses are shown in Figure 4.9. The eight consecutive steps are summarized as follows:

$$\begin{cases} v_{d,1} & = 5m/s \\ v_{d,2} & = 0m/s \end{cases} \rightarrow \begin{cases} v_{d,1} & = 5m/s \\ v_{d,2} & = 5m/s \end{cases} \rightarrow \begin{cases} v_{d,1} & = 0m/s \\ v_{d,2} & = 5m/s \end{cases} \rightarrow \begin{cases} v_{d,1} & = -5m/s \\ v_{d,2} & = 5m/s \end{cases}$$
$$\rightarrow \begin{cases} v_{d,1} & = -5m/s \\ v_{d,2} & = 0m/s \end{cases} \rightarrow \begin{cases} v_{d,1} & = -5m/s \\ v_{d,2} & = -5m/s \end{cases} \rightarrow \begin{cases} v_{d,1} & = 0m/s \\ v_{d,2} & = -5m/s \end{cases} \rightarrow \begin{cases} v_{d,1} & = 0m/s \\ v_{d,2} & = 0m/s \end{cases}$$

Similar to trajectory 1, $v_{d,3}$ were maintained at zero for the entirety of trajectory 2.

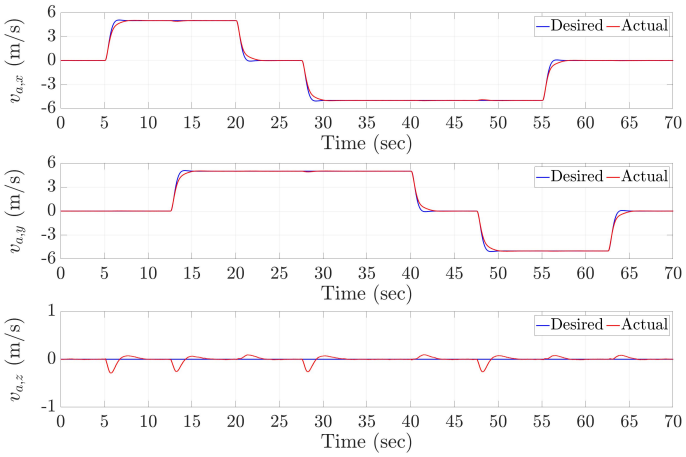
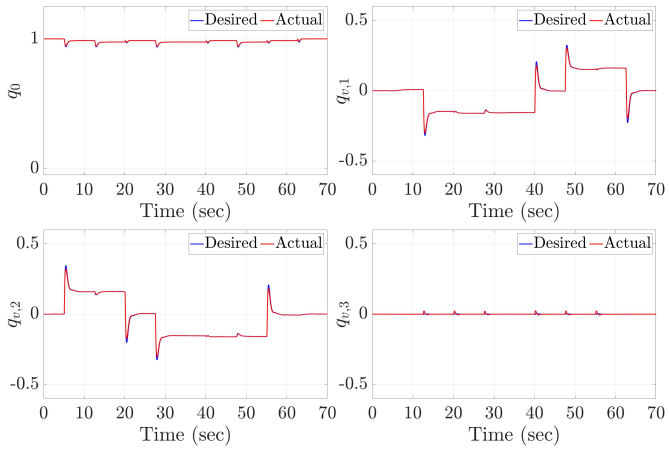
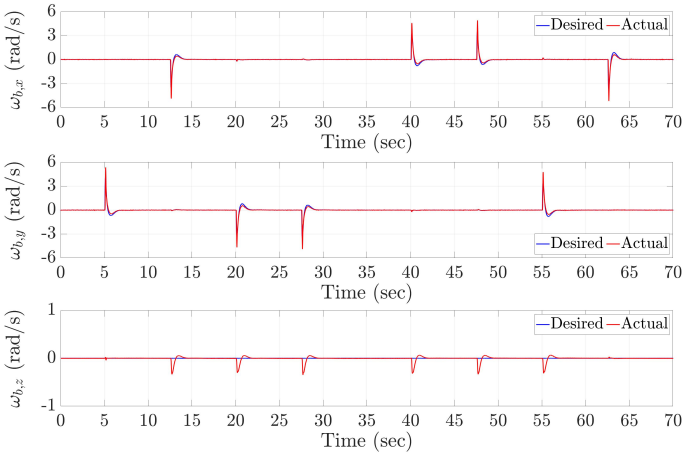


Figure 4.8: Velocity Response of Sinusoidal Trajectory 2



(a) Quaternion Attitude

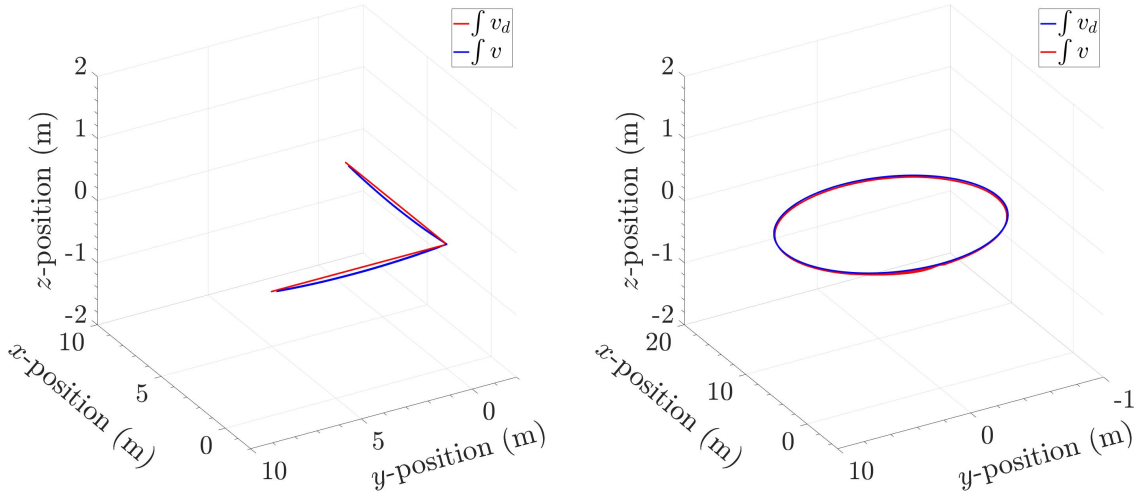


(b) Angular Velocity

Figure 4.9: Attitude Response of Sinusoidal Trajectory 2

4.2.2 Sinusoidal Response

Similar to the step trajectories, the following section demonstrates the performance of the quadrotor under two sinusoidal trajectories. Sinusoidal Trajectory 1 involves moving the quadrotor forward and backwards, or sideways, in one direction. The second trajectory simulates a circular motion with two simultaneous trajectories in both the x and y directions. Both the first and second trajectory had a fixed desired z-velocity of zero. Figure 4.10 is a visual representative of the quadrotor position when it follows the two sinusoidal trajectories.



(a) Trajectory 1 (b) Trajectory 2

Figure 4.10: Sinusoidal Trajectories

4.2.2.1 Sinusoidal Trajectory 1

Figures 4.11 and 4.13 displays the step velocity response of the quadrotor when tracking sinusoidal trajectory 1; each figure illustrates the quadrotor tracking sinuosidal velocity inputs in the x and y directions. The corresponding attitude and angular velocity responses are shown in Figures 4.5 and 4.7. The figures show the response of the quadrotor while tracking a sine trajectory with a period of 1.5π and an amplitude of 3 m/s for both the x and y directions.

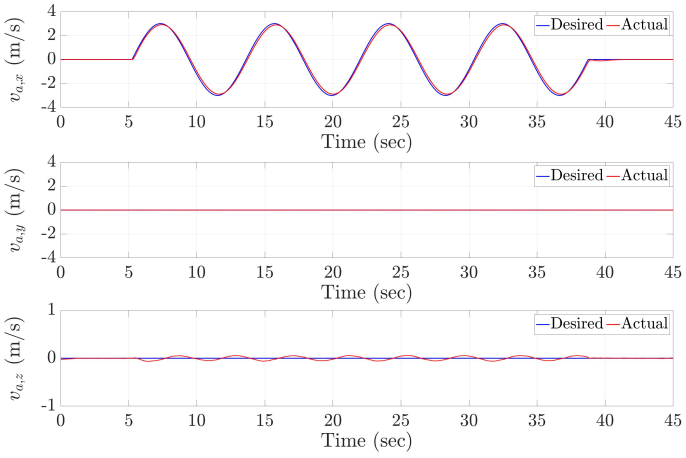
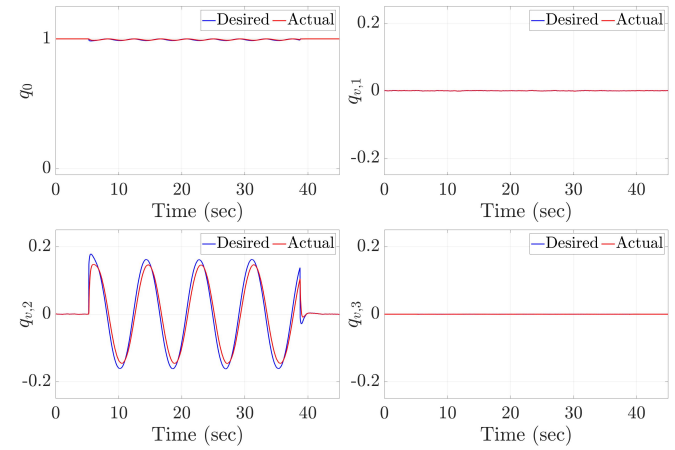
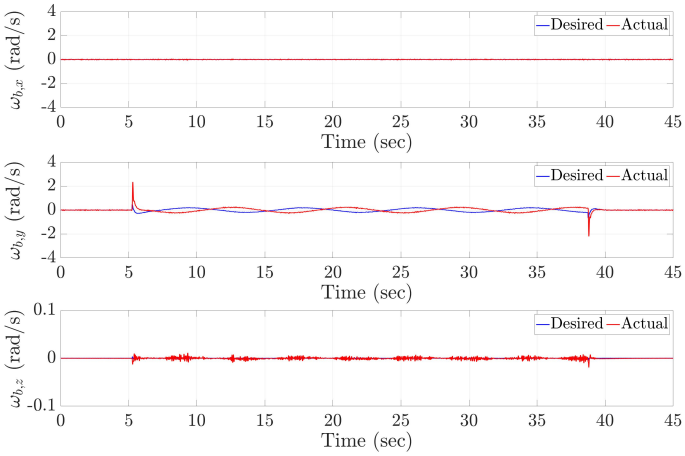


Figure 4.11: Velocity Response of Sinusoidal Trajectory 1 in the y-Direction



(a) Quaternion Attitude



(b) Angular Velocity

Figure 4.12: Attitude Response of Sinusoidal Trajectory 1 in the x-Direction

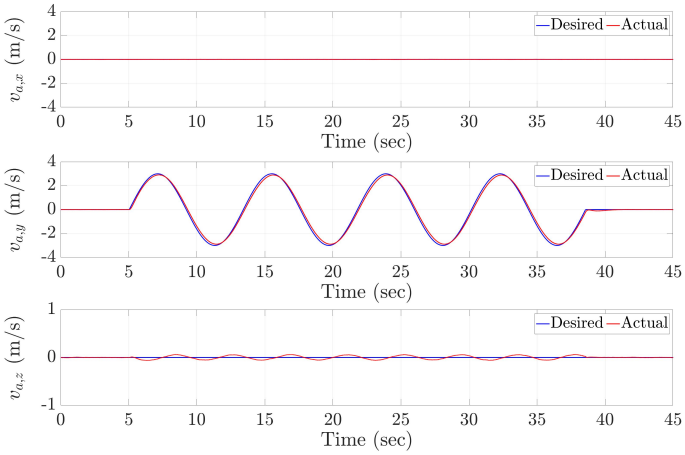
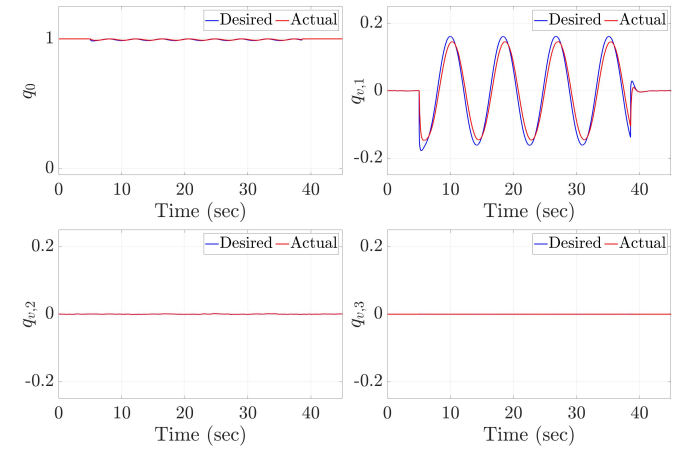
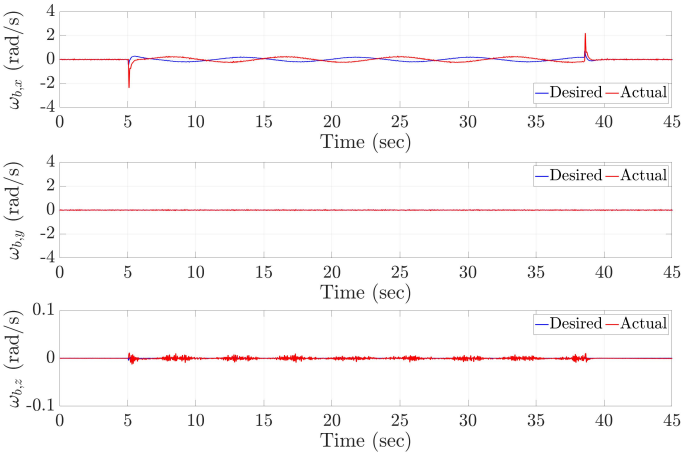


Figure 4.13: Velocity Response of Sinusoidal Trajectory 1 in the y-Direction



(a) Quaternion Attitude



(b) Angular Velocity

Figure 4.14: Attitude Response of Sinusoidal Trajectory 1 in the y-Direction

4.2.2.2 Sinusoidal Trajectory 2

Figure 4.15 displays the velocity response of the quadrotor following sinusoidal trajectory 2, a circular trajectory; the corresponding attitude response can be found in Figure 4.16. The circular trajectory consists of a sine wave and cosine wave, both with an amplitude of 2 m/s and a period of 0.5 π, in the x and y directions, while maintaining the velocity in the z direction at zero. The trajectory can be expressed as follows:

$$\begin{cases} v_{d,1} &= 2\sin 0.25t \\ v_{d,2} &= 2\cos 0.25t. \end{cases}$$

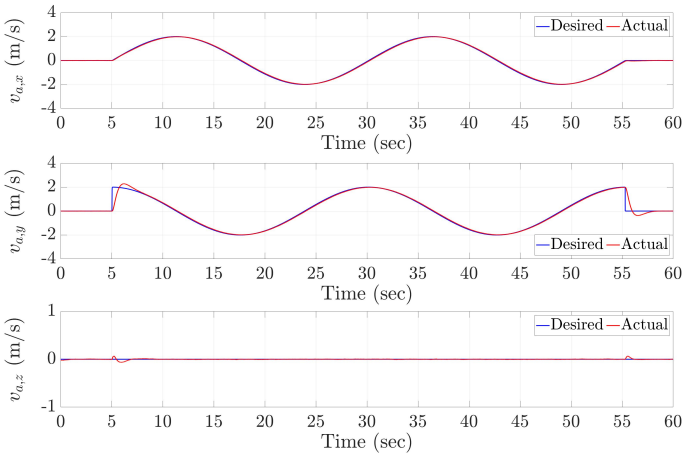
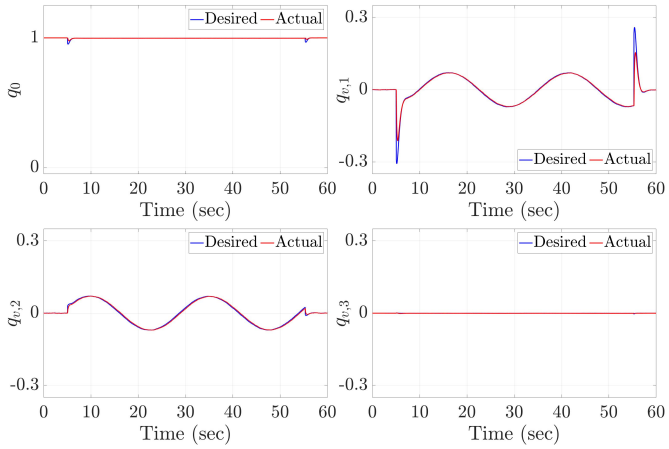
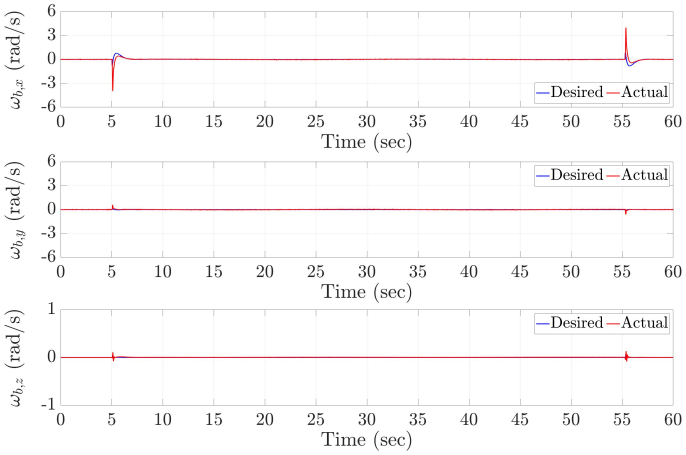


Figure 4.15: Velocity Response of Sinusoidal Trajectory 2



(a) Quaternion Attitude



(b) Angular Velocity

Figure 4.16: Attitude Response of Sinusoidal Trajectory 2

4.2.3 *Flipping Motion*

The final tested trajectory is a velocity trajectory designed to flip the quadrotor $360°$ in any desired direction. The velocity trajectory for the $360°$ flip motion is as follows:

$$\begin{cases} v_{d,1} = & \check{r}_1 \frac{3g}{8\pi}(\cos\left(\frac{8}{3}\pi t\right) - 1) \\ v_{d,2} = & \check{r}_2 \frac{3g}{8\pi}(\cos\left(\frac{8}{3}\pi t\right) - 1) \\ v_{d,3} = & \frac{g}{2}(\frac{3}{4\pi}\sin\left(\frac{8}{3}\pi t\right) - t) \end{cases} \tag{4.6}$$

where $\check{r} = [\check{r}_1 \;\; \check{r}_2]^T$ is a normalized vector that determines the direction of flip. After one cycle of the $360°$, v_d returns to zero to complete the flip and stabilize. Note that instead of the third order filter used for normal flight, the velocities, and its derivatives, of the flipping motion was only lightly filtered with a first order low pass filter as the third order filter was too slow to successfully achieve a $360°$ flip. Figure 4.17 shows the quadrotor tracking a $360°$ flip velocity trajectory. The flipping motion is done consecutively in 3 different directions: x ($\check{r} = [1 \;\; 0]^T$), y ($\check{r} = [1 \;\; 0]^T$), and $-x/-y$ ($\check{r} = [-\frac{1}{\sqrt{2}} \;\; -\frac{1}{\sqrt{2}}]^T$) directions.

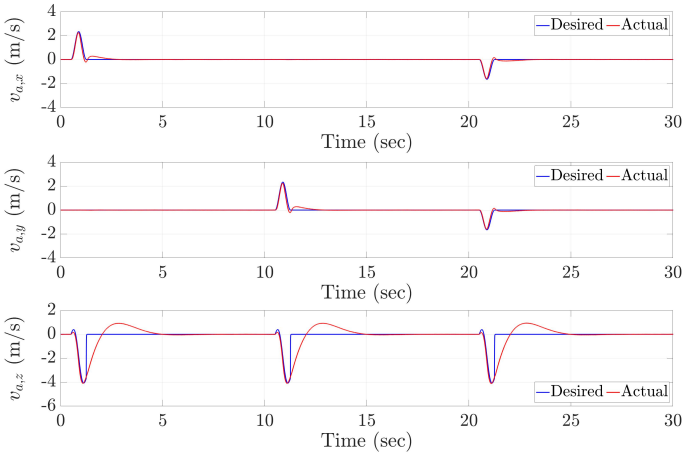
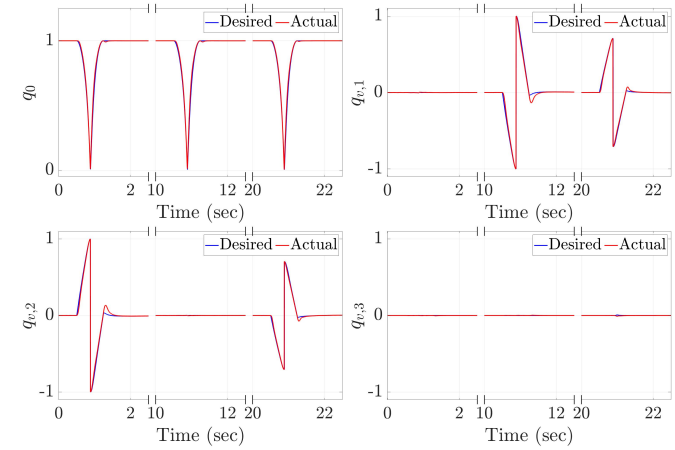


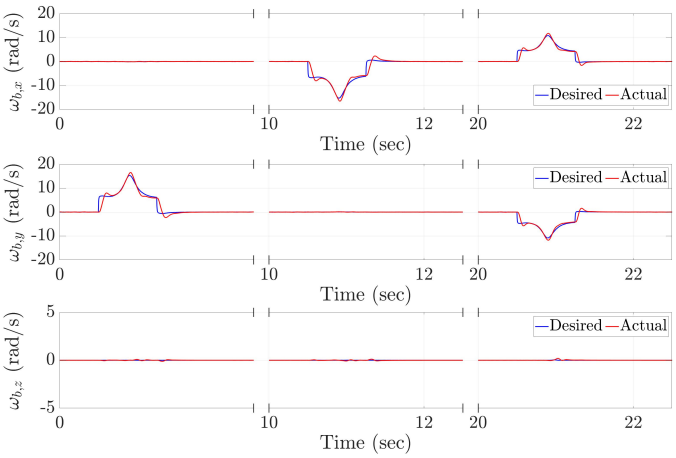
Figure 4.17: Velocity Response of Flipping Trajectory

Figure 4.18 displays the quaternion attitude and angular velocity responses during the flip. The result of the consecutive flips can be seen more intuitively in Figure 4.19,

which shows the Euler Angles converted from the quaternion attitude during each flip.



(a) Quaternion Attitude



(b) Angular Velocity

Figure 4.18: Attitude Response of Flipping Trajectory

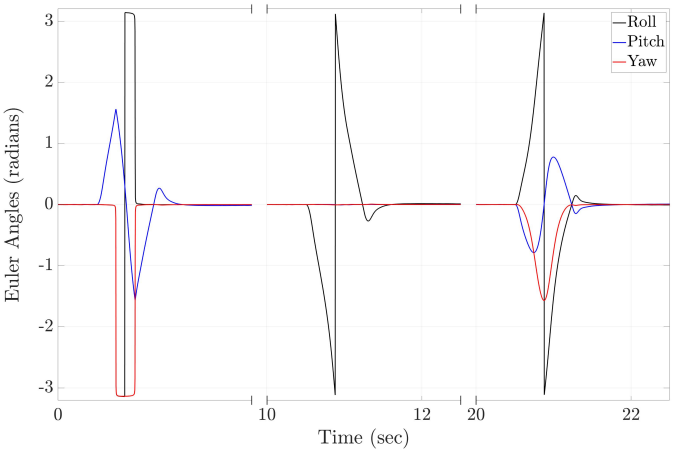
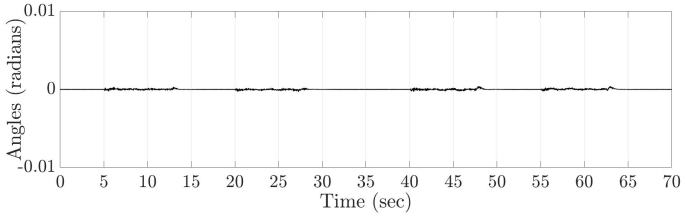


Figure 4.19: Euler Angles during Flipping Motion

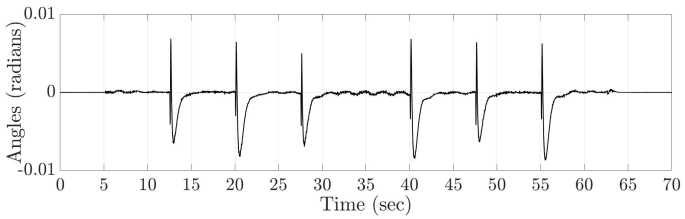
4.3 Discussion

4.3.1 Difference between u_d and $R_q^T u_d$ ($\bar{\gamma}$)

According to Proposition 3.3 in Section 3.6.3, the difference between finding the quaternion attitude error $\tilde{q}$ the two methods described in Equation 3.44 can be given by a pure rotation around the e_z axis, $\bar{\gamma}$. Figures 4.20 and 4.21 each shows the $\bar{\gamma}$ values computed for the two step and sinusoidal trajectories. The first step and sinusoidal trajectories are both uni-direction trajectories (i.e., rotation while tracking the trajectories only occur along 1 axis on the $e_x e_y$ plane); in other words, the z-component of the rotation axis of $u_d \rightarrow u$ is kept close to zero. As a result, as shown in Figures 4.20a and 4.21a, $\bar{\gamma}$ is kept close to zero for step and sinusoidal trajectory 1. On the other hand, the second step and sinusoidal trajectories include rotations in which the z-component of the rotation axis of $u_d \rightarrow u$ is not zero; hence, according to Figures 4.20b and 4.21b, $\bar{\gamma}$ does not maintain close to zero as much as the first step and sinusoidal trajectories do. Even so, the $\bar{\gamma}$ value is relatively small ($|\bar{\gamma}| < 0.01$); therefore, the difference between the two methods of obtaining $\tilde{q}$ is negligible. In fact, the method of finding $\tilde{q}$ with $R_q^T u_d$ is preferred as it eliminates the need to consider the singularity that occurs when $u_d = [0 \;\; 0 \;\; -1]^T$ as explained in Section 3.3.

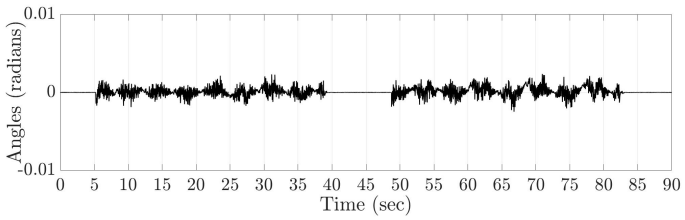


(a) Step Trajectory 1

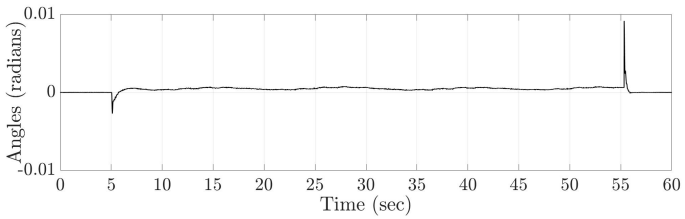


(b) Step Trajectory 2

Figure 4.20: $\bar{\gamma}$ of Step Trajectories



(a) Sinusoidal Trajectory 1



(b) Sinusoidal Trajectory 2

Figure 4.21: $\bar{\gamma}$ of Sinusoidal Trajectories

4.3.2 $\hat{\gamma}$ Approximation

As discussed previously, the $\hat{\gamma}$ value needed during simulation was approximated using $\check{\gamma}$ with the method detailed in Equation 4.3 in Section 4.1.2. The approximation was done to ensure that the quaternion used during control is smooth continuous so that the resulting control inputs are nonsingular. Figure 4.22 shows the comparison between the $\hat{\gamma}$ values each obtained with Equation 3.28, Equation 4.3 with $\check{\gamma} = \hat{\gamma}_d$ and Equation 4.3 with $\check{\gamma} = \hat{\gamma}_{(n-1)}$ where $\hat{\gamma}_d$ is the desired $\hat{\gamma}$ angle and $\hat{\gamma}_{(n-1)}$ is the $\hat{\gamma}$ from one previous time step during the flipping trajectory shown in Figure 4.17. According to the results, the two approximated values, each shown in red and blue, are smooth and continuous unlike $\hat{\gamma}$ found without approximation. In addition, setting $\check{\gamma} = \hat{\gamma}_{(n-1)}$ results in an approximation that tracks the actual $\hat{\gamma}$ values more closely; while setting $\check{\gamma} = \hat{\gamma}_d = 0$ results in an approximation that is kept closer to zero. The results are much expected as Equation 4.3 essentially is a smooth transition between the actual $\hat{\gamma}$ and the selected $\check{\gamma}$ whenever singular.

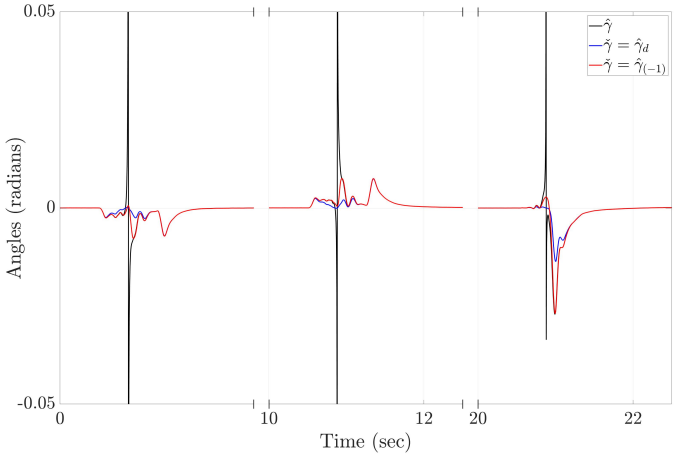


Figure 4.22: $\bar{\gamma}$ Approximation of Flipping Motion

4.3.3 Singularity Free Control Input

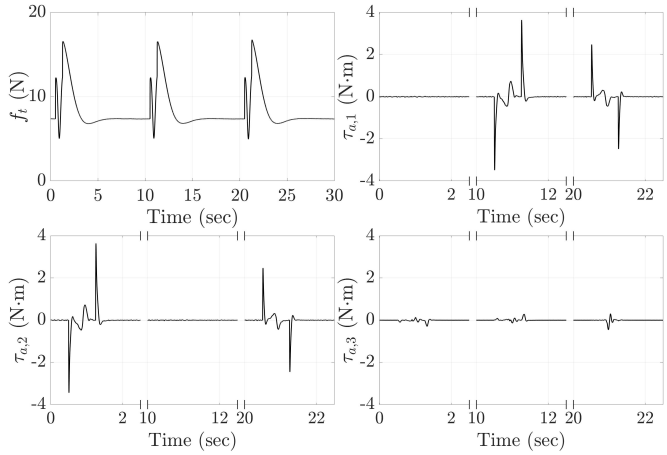


Figure 4.23: Control Inputs during Flipping Motion

Figure 4.23 shows the 4 control inputs, f_t and τ_a during the tested flipping motion. As shown in the figure, the control inputs are smooth and non-singular since most of the singularities are dealt with.

CHAPTER 5
IMPLEMENTATION

The following sections discuss the implementation of the simulated controller. The controller was implemented in an outdoors environment and the reference velocity inputs were passed to the controller via a RC transmitter and receiver. As with simulations, the control inputs were computed assuming $\bar{\gamma}$ stays close to zero; in other words, the attitude error $\tilde{q}$ were computed from $R_q^T u_d$.

5.1 Implementation Details

5.1.1 Implementation Hardware Properties

The quadrotor used during implemented was a commercial quadrotor the details of the hardware can be found in Appendix C. The physical parameters of the hardware can be found in Table 5.1. The inertial properties of the quadrotor were estimated with a 3D CAD model of the actual hardware. The lift coefficient of the propellers was estimated by looking into the thrust ratio needed to hover the quadrotor. As mentioned before, the open-source software used to build the controller code normalizes all torque and force values with the square of the maximum rotor angular velocity, Ω_{max}. If m, g, and Ω_{max} is known, the lift coefficient b can then be found with Equation 4.5. The drag coefficient κ, on the other hand, was chosen so that it was similar in magnitude to other propellers with similar size and blade pitch angle.

5.1.2 Implementation Gains

Table 5.2 shows the control gains used during implementation of the controller. Note that an additional lead compensation, in terms of $\dot{\tilde{\omega}}_r$, was added to the control law

Table 5.1: Model Parameters of Simulation Aircraft

Property	**Parameter**	**Value**
Mass in kg	m	1.207
Moment of Inertia about x-axis in $kg \cdot m^2$	I_{xx}	1.267×10^{-2}
Moment of Inertia about y-axis in $kg \cdot m^2$	I_{yy}	1.239×10^{-2}
Moment of Inertia about z-axis in $kg \cdot m^2$	I_{zz}	2.359×10^{-2}
Propeller Lift Coefficient	b	1.031×10^{-5}
Propeller Drag Coefficient	κ	2×10^{-7}
Arm Length in m	l	0.3

(Equation 3.6) during implementation. The additional term compensates for sensor lag and adds additional damping to the transient response. For clarity, let $K_{\omega,p} = k_{\omega,p} I_{3\times3} \in \mathbb{R}^{3\times4},\ \ k_{\omega,p} > 0$ be the original K_ω, and $K_{\omega,d} = k_{\omega,p} I_{3\times3} \in \mathbb{R}^{3\times4},\ \ k_{\omega,d} > 0$ be the gain of the added term. The new attitude control law, with the added gain, can then be expressed as follows:

$$\tau_a = I_f \dot{\omega}_r + \omega_r \times I_f \omega + G_a - K_{\omega,p} \tilde{\omega}_r - K_{\omega,d} \dot{\tilde{\omega}}_r - K_q \tilde{q}. \tag{5.1}$$

5.1.3 Gyroscopic Torque Feedforward Term

The gyroscopic torque feedfoward term needed in the control law defined in Equation 5.1 were estimated as the actual rotation velocities of the rotors were challenging to acquire during flight. The angular velocities of the rotors at different PWM values were acquired beforehand and the relationship between the two variables is used during flight to compute the gyroscopic term. The experimentally measured relationship between the two variables are shown in Figure 5.1. The data was fitted with a linear trend, and the gyroscopic torque was computed with a rotor inertia of $I_r = 1.866 \times 10^{-4}$ approximated

Table 5.2: Controller Gains, Implementation

Gain Description	**Parameter**	**Value**
Linear Velocity	K_v	$\begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 2 \end{bmatrix}$
Linear Position	K_x	$\begin{bmatrix} 2 & 0 & 0 \\ 0 & 2 & 0 \\ 0 & 0 & 3 \end{bmatrix}$
Quaternion	K_q	$[0_{3\times1} \;\; 2.3333 I_{3\times3}]$
ω_r Proportional	$K_{\omega,p}$	$0.3333 I_{3\times3}$
ω_r Derivative	$K_{\omega,d}$	$0.0139 I_{3\times3}$
Variable Gain 1	λ_q	0.0208
Variable Gain 2	λ_u	0.0521

with the size and weight of each of the propellers.

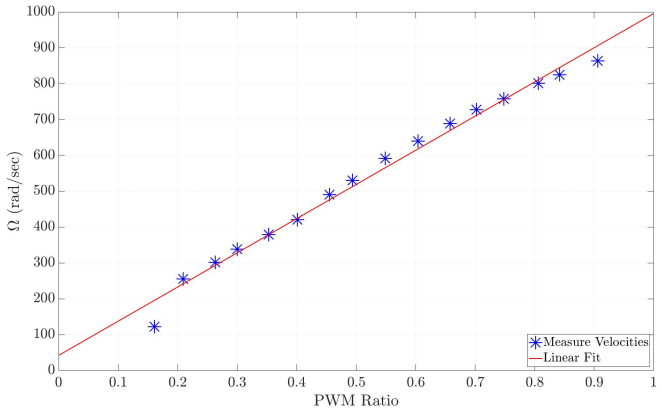


Figure 5.1: Relationship between PWM ratio and Rotor Angular Velocities

5.2 Implementation Results

This section presents the performance of the controller implemented on a commercial quadrotor under various trajectories; similar to the simulation results, the section will

first present the controller performance under simple trajectories and proceed to present the responses of the quadrotor under more involved trajectories.

5.2.1 Stabilization

Figure 5.2 shows the velocity response of the quadrotor while stabilizing around zero velocity (i.e., $v_d = [0 \;\; 0 \;\; 0]^T$); the attitude responses, for both the quaternion attitude and angular velocity, can be found in Figure 5.3.

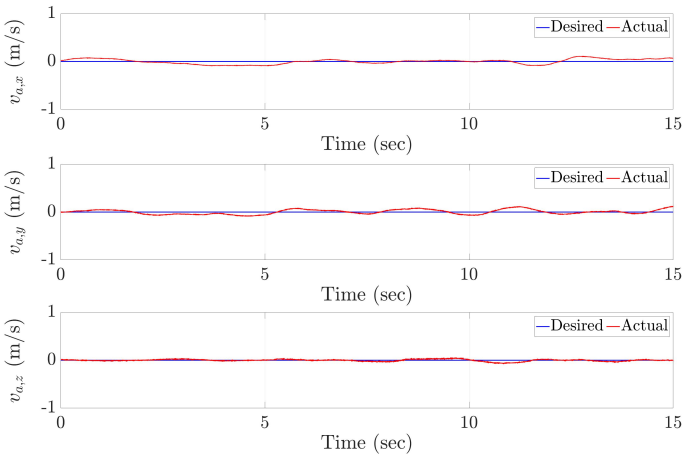
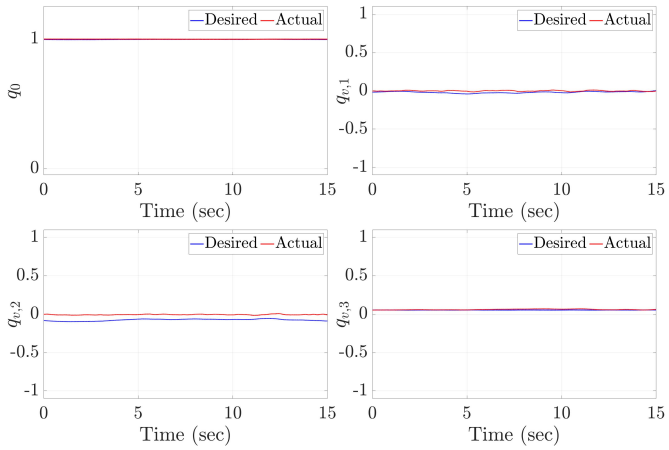
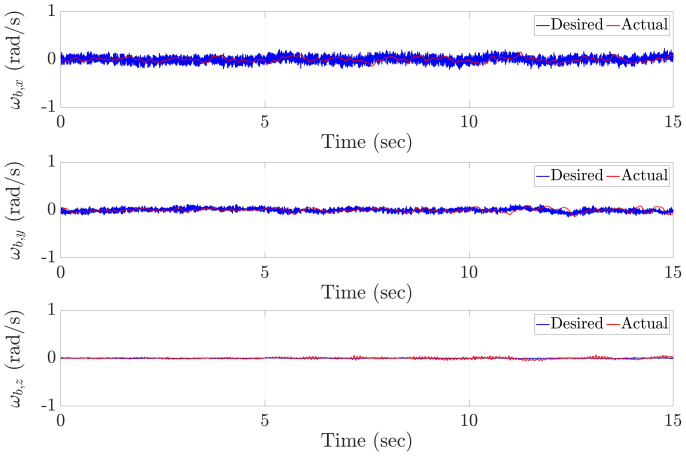


Figure 5.2: Velocity Response during Stabilization



(a) Quaternion Attitude



(b) Angular Velocity

Figure 5.3: Attitude Response during Stabilization

5.2.2 Step Trajectory 1

Figures 5.4 and 5.6 display the velocity responses of a quadrotor under a uni-directional step trajectory, in both the x and y directions, as described in Figure 4.3; Figures 5.5 and 5.7 show the corresponding attitude responses of the quadrotor. As shown in Figure 5.4, and in Figure 5.6, the amplitude of each step is 5 m/s. As before, the desired trajectories show the filtered reference trajectories instead of the raw 5 m/s step jumps.

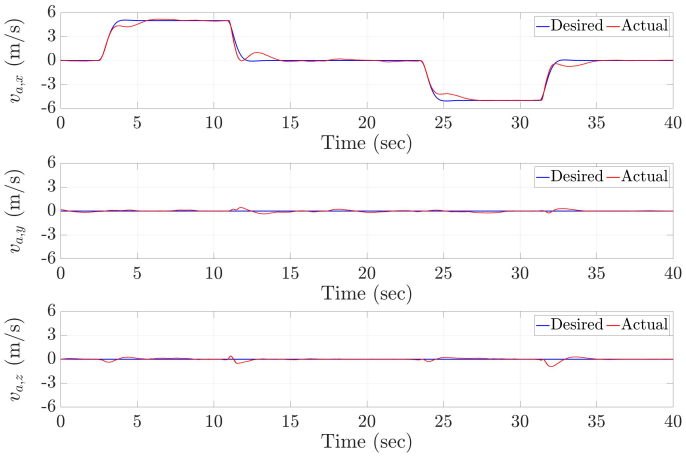
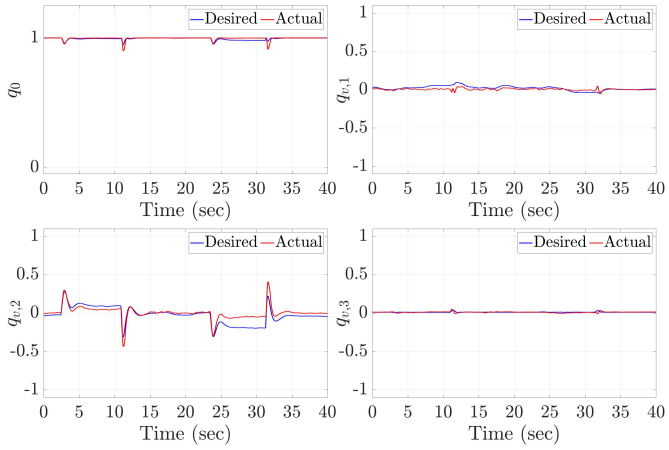
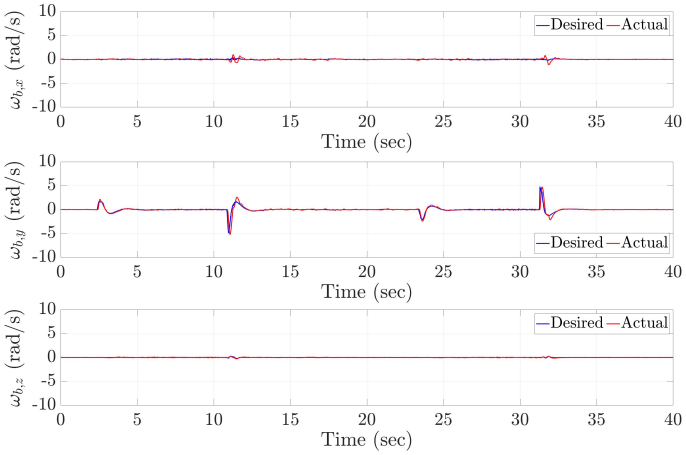


Figure 5.4: Velocity Response of Step Trajectory 1 in x Direction



(a) Quaternion Attitude



(b) Angular Velocity

Figure 5.5: Attitude Response of Step Trajectory 1 in x Direction

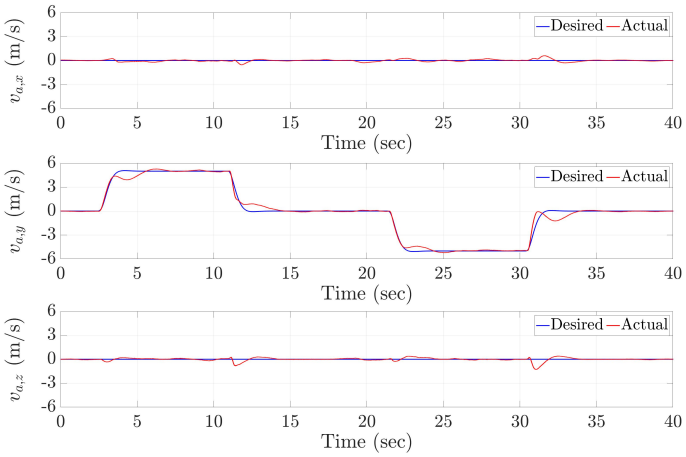
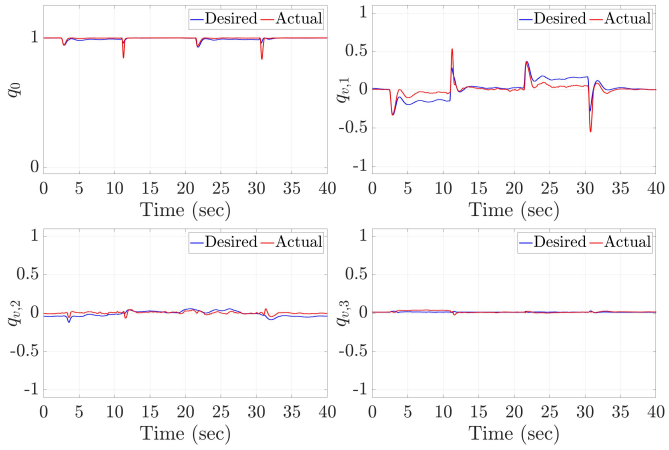
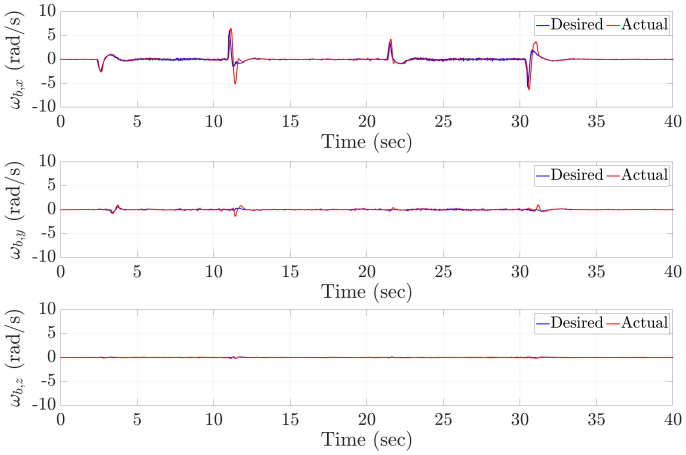


Figure 5.6: Velocity Response of Step Trajectory 1 in y Direction



(a) Quaternion Attitude



(b) Angular Velocity

Figure 5.7: Attitude Response of Step Trajectory 1 in y Direction

5.2.3 *Sinusoidal Trajectory 1*

Figures 5.8 and 5.10 display the velocity responses of a quadrotor under a uni-directional sinusoidal trajectory, in both the x and y directions, as described by Sinusoidal Trajectory 1 shown in Figure 4.10; Figures 5.9 and 5.11 show the corresponding attitude responses of the quadrotor. The sinusoidal properties of the trajectory are the same as simulations; the trajectory was a sine wave with a period of 1.5π and an amplitude of 3 m/s.

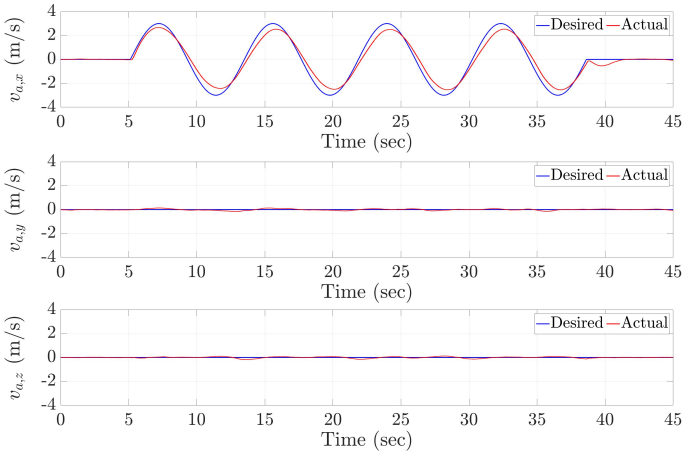
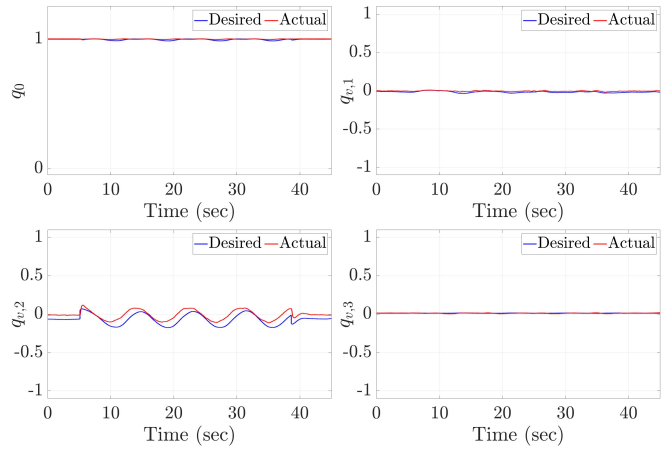
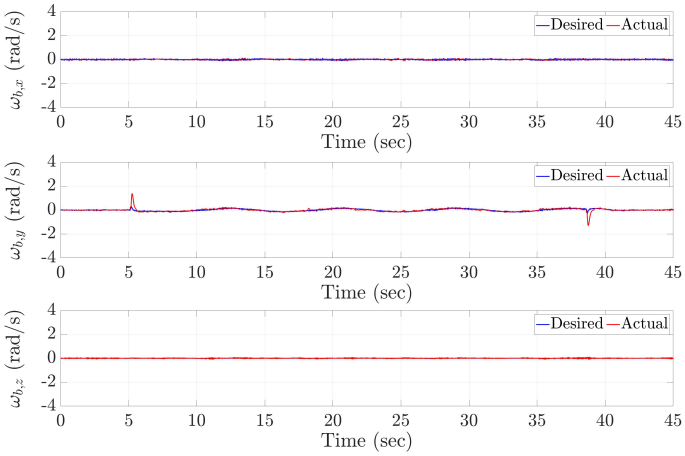


Figure 5.8: Velocity Response of Sinusoidal Trajectory 1 in x Direction



(a) Quaternion Attitude



(b) Angular Velocity

Figure 5.9: Attitude Response of Sinusoidal Trajectory 1 in x Direction

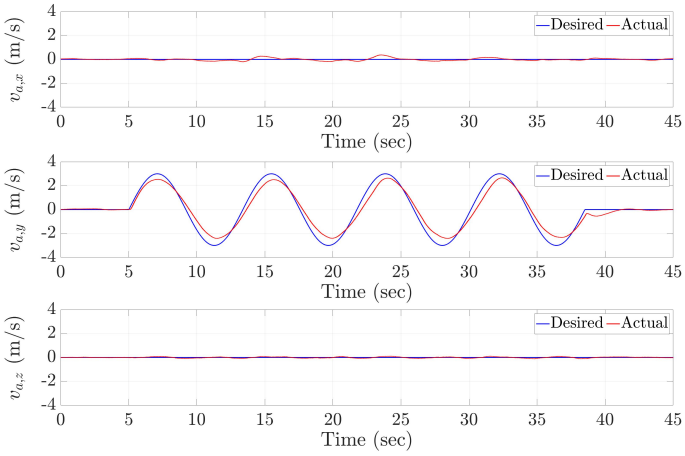
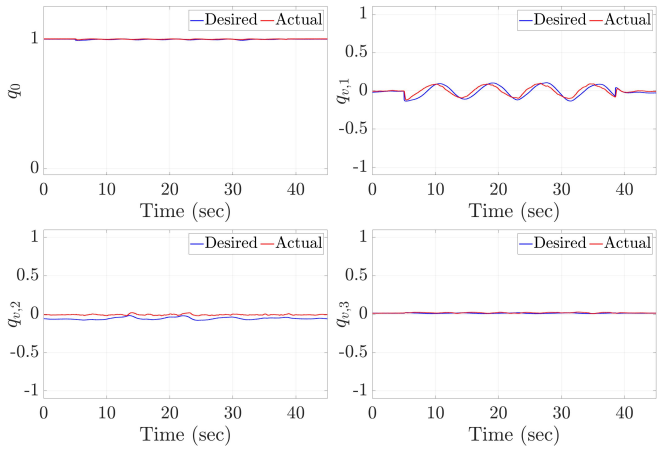
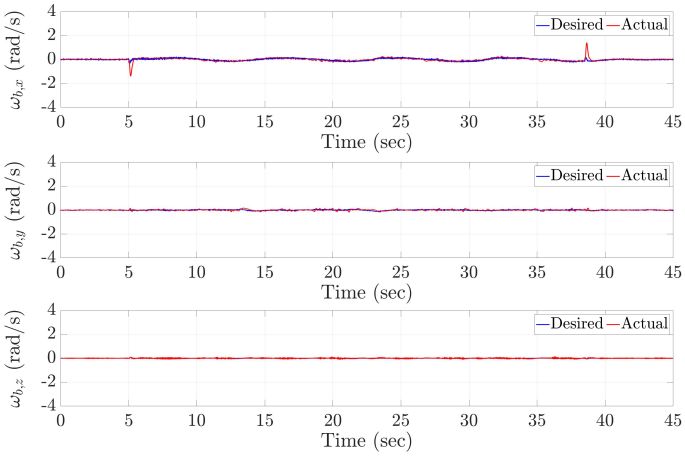


Figure 5.10: Velocity Response of Sinusoidal Trajectory 1 in y Direction



(a) Quaternion Attitude



(b) Angular Velocity

Figure 5.11: Attitude Response of Sinusoidal Trajectory 1 in y Direction

5.2.4 Sinusoidal Trajectory 2

Figures 5.12 and 5.13 each show the velocity and attitude responses of the quadrotor under the second sinusoidal trajectory as described in Figure 4.10. As shown in Figure 5.12, the quadrotor tracks a sine trajectory with a period of 0.5π and an amplitude of 2 m/s in the x direction, while simultaneously tracking a cosine trajectory with the same period and amplitude in the y-direction.

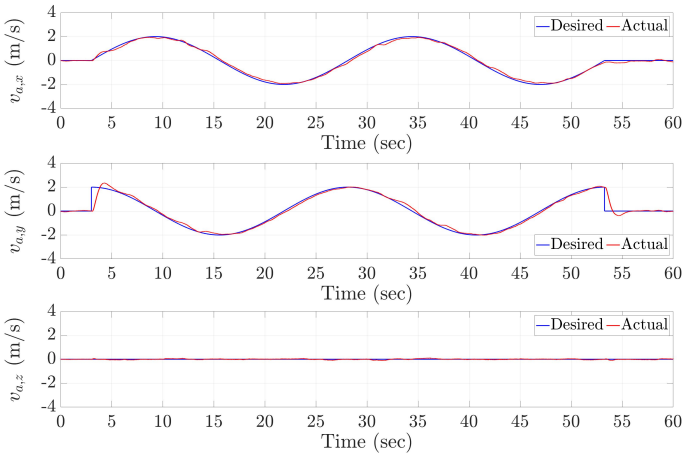
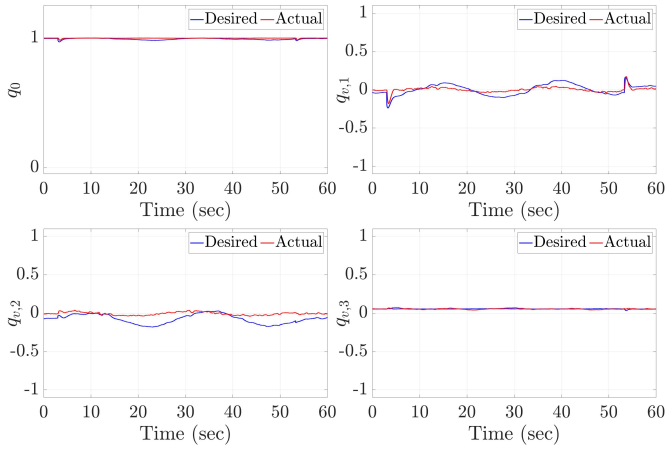
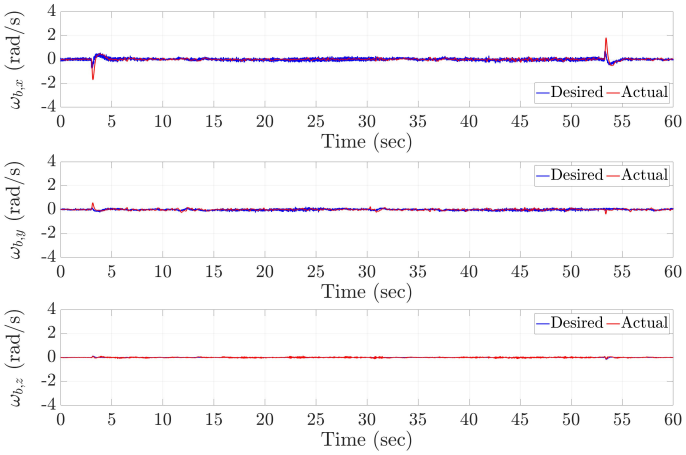


Figure 5.12: Velocity Response of Sinusoidal Trajectory 2



(a) Quaternion Attitude



(b) Angular Velocity

Figure 5.13: Attitude Response of Sinusoidal Trajectory 2

5.2.5 *Flipping Motion*

Figures 5.14 and 5.15 display the velocity and attitude response to a flipping motion described by Equation 4.6. As with simulations, three consecutive flips are tested all in the x, y, and $-x/-y$ direction. Figure 5.16 additional shows the Euler Angles of quadrotor during the flip to give better, and more intuitive, illustration of the flipping motion.

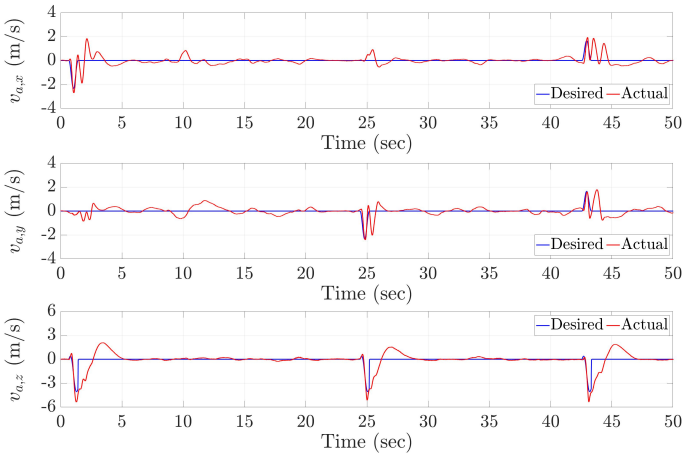
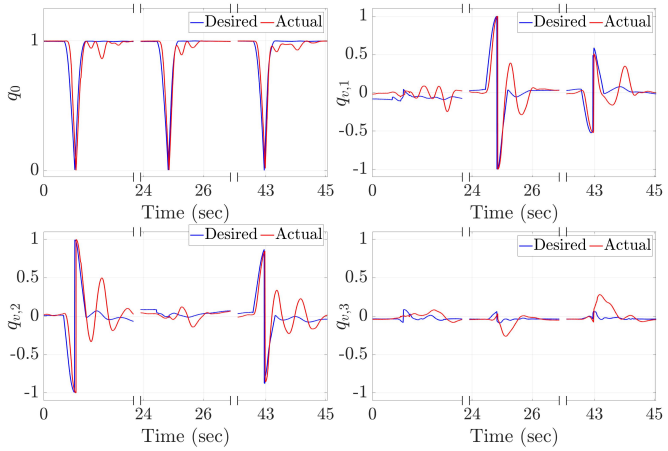
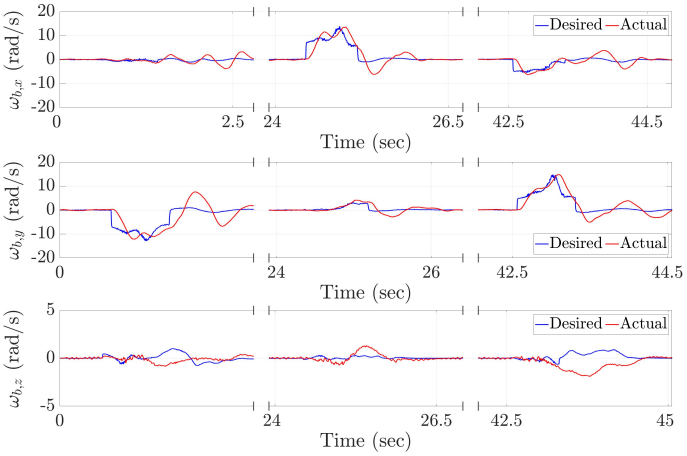


Figure 5.14: Velocity Response of Flipping Trajectory



(a) Quaternion Attitude



(b) Angular Velocity

Figure 5.15: Attitude Response of Flipping Trajectory

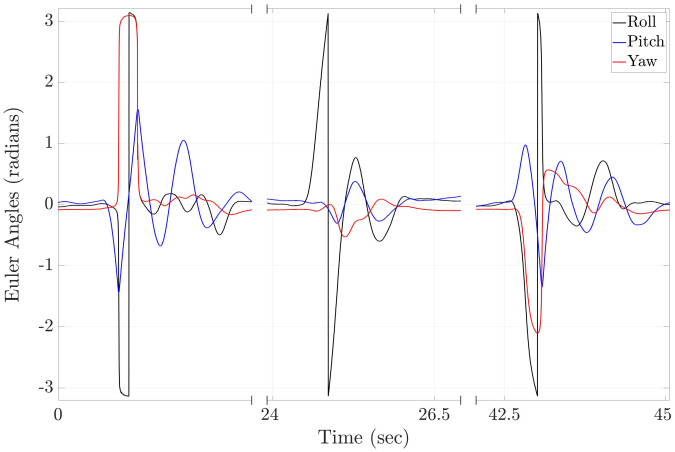


Figure 5.16: Euler Angles during Flipping Motion

5.3 Discussion

The following implementation results demonstrate the controller stabilizing at zero reference, and tracking step, sinusoidal, and a 360° flipping trajectories. All testing of the controller were done on a moderately windy day (10 - 15 km/h). According to Figures 5.2 and 5.3, the quadrotor was able to stabilize at 0 m/s without any oscillation in either the velocity, attitude, or angular velocity responses, which shows the robustness of the controller. Note that during stabilization there was an offset in $q_{v,3}$; this offset was mainly due to the calibration imbalance of the two rotors controlling the pitch of the aircraft. Even so, the integral control included in the outer velocity controller accounts for this imbalance allowing the overall control to be stable without any steady-state velocity error. The subsequent responses of the quadrotor tracking step and sinusoidal trajectories (Figures 5.4, 5.6, 5.8 and 5.10) demonstrate similar results to simulation (Figures 4.4, 4.6, 4.11 and 4.13) even in the presence of external disturbances such as the effects of wind and additional sensor noise/delay. The implemented flip motion (Figure 5.14) was also able to track the trajectory while keeping the quaternion error relatively close to unit quaternion (Figure 5.17). The larger error present during implementations when compared with the simulation results can largely be explained by the attitude and angular velocity sensor lags. The attitude sensor lag was around 0.25 seconds, which is significant considering that the whole duration of each of the flipping motion was 0.75 seconds. This large difference in sensor lag also explains why the controller needed an additional feedback term in terms of $\dot{\tilde{\omega}}_r$ in the attitude control law as explained in Section 5.1.2. Nevertheless, the controller was still able to perform the flip and stabilize back at equilibrium in spite of the lag.

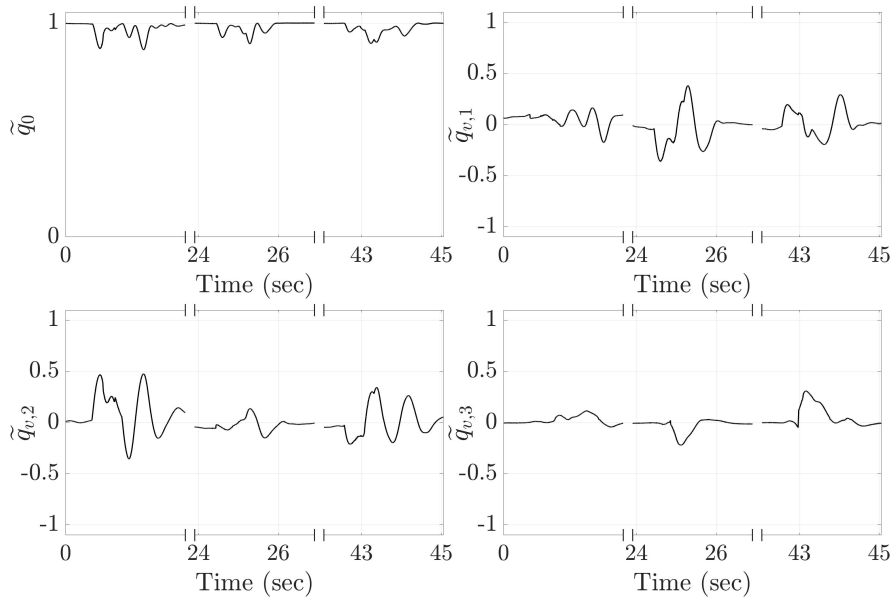


Figure 5.17: Attitude Error during 360° Flip

5.3.1 Controller Gains

According to both simulations and implementation, the results demonstrate robust and accurate control for all trajectories using the same controller gains. This is largely due to the modification made to ω_d as defined in Equation 3.3. According to Equation 3.22, the gain Λ_q varies with $\|u_d\|$. $\|u_d\|$ is generally larger for more complicated trajectories which require larger changes in thrust. Hence, Λ_q, according to Equation 3.22, adds more proportional feedback which accounts for the larger attitude changes present during more ambitious trajectories. On the other hand, for relatively more simple trajectories with smaller $\|u_d\|$ the contribution of Λ_q to the attitude control torque is not as large.

5.3.2 Practicality of Controller

At this point, it is important to discuss the actual practicality of the designed controller. Both the simulation and implementation results show that controller can be effectively used to track simple to complicated trajectories while the attitude of the quadrotor is well-conditioned. However, even if the controller is well designed for the specific purpose;

it does not necessarily mean that purpose itself is desirable, or practical. Such practicality, or dynamic feasibility, has been discussed in many literature; in fact, many papers focus on designing dynamically feasible trajectories [31, 32, 49]. As mentioned previously, the advantage of this particular controller has over other locally stabilizing controller is in that the specific controllers allows the quadrotor attitude to achieve any attitude configuration when needed. However, this advantage also suggests that such extreme attitude configuration may be achieved in situations wherein it is more advantageous to the overall control that the attitude configuration remains within the upper region; such globality of the controller may be undesirable for certain situations. Hence, it is important that the trajectories are designed accordingly. For instance, the reference position, or velocity, trajectory may be filtered, as explained in Section 4.1.1, to prevent the quadrotor from executing desired trajectories with too much aggression. In other words, the controller would be best fit to be used with appropriately designed trajectories that fit the particular purpose.

CHAPTER 6
CONCLUSION

The quadrotor is an interesting system to control due to many of its inherent characteristics. Controlling quadrotors may be challenging due to its nonlinearities. Quadrotors during flight do not have any external forces other than the gravitational force which makes it simple to model but difficult to achieve stable control. Globally stable control of the quadrotor is especially tricky to achieve as it requires the controllers to guarantee stability at any point of attitude for any desired velocity or position trajectory with any initial state error conditions.

Due to current mathematical limitations, singularities are bound to occur, specifically in the attitude of the quadrotor, no matter which attitude convention is used. The fact that the control scheme is required to to couple the three degrees of freedom in the 3D rotational space with the three degrees of freedom in the 3D translational space makes globally stable control even complicated as additional singularities are introduced. Many quadrotor control schemes limit the rotational orientation to avoid such singularities; however, these limitations introduces difficulties when tracking more ambitious trajectories, such as flipping motions or stabilizing after free fall.

The thesis investigates such singularities and difficulties in establishing global control of such aircrafts and presents simple and realistic solutions given the limitations of current technologies. A globally PI-PD control scheme is presented which effectively achieves globally stable control of the quadrotor, in the sense that the error of the quadrotor states exponentially converges to zero regardless of how large the initial state error is, once all singularities are addressed. As shown in both the simulation and implementation results, the presented control scheme allows the quadrotor to follow ambitious trajectories, which require robust and global control of the quadrotor, smoothly and accurately even with the

introduction of disturbances such as wind or calibration errors. In addition, the results demonstrate that the presented controller is equally robust while tracking ambitious trajectories with large changes in attitude as it is tracking simpler trajectories without changing any control parameters.

Although the feasibility of the controller itself have been shown, additional consideration of dynamic feasibility while using the designed controller must be made. Due to the globality of the controller, the quadrotor may be forced to track trajectories with unnecessarily ambitious attitude maneuvers. Hence, improvement on this work may involve designing a dynamically feasible trajectory planning strategy, which would operate with the proposed controller; such trajectory planning strategies may allow the quadrotor to undergo agile maneuvers with large attitude changes only when necessary.

Appendices

APPENDIX A

ADDITIONAL THEOREMS AND TECHNICAL LEMMAS

Lemma A.1. *The time derivative of the term* $(q-1)^T(q-1)$ *may be expressed as:*

$$\frac{d}{dt}\left[(q-1)^T(q-1)\right] = \bar{\omega}_b^T q. \tag{A.1}$$

Proof. Keeping in mind of Equation 2.8, the considered expression can be simplified as:

$$\begin{aligned}(q-1)^T(q-1) &= q^T q - 2q_0 + 1 \\ &= 2(1-q_0)\end{aligned}$$

where q_0 is the scalar part of the quaternion. With Equation 2.11, the time derivative of q_0 can be obtained by:

$$\dot{q}_0 = -\frac{1}{2}\bar{\omega}_b^T q_v. \tag{A.2}$$

Then, with Equation A.2, Equation A.1 can be shown to true as follows:

$$\frac{d}{dt}\left[(q-1)^T(q-1)\right] = -2\dot{q}_0 = \bar{\omega}_b^T q.$$

□

Lemma A.2. *The norm of the input error* $\tilde{u}$ *satisfies* $\|\tilde{u}\| \leq 2\|u_d\|\,\|\tilde{q}_v\|$.

Proof. Consider $\|\tilde{u}\|^2$, which is given by:

$$\|\tilde{u}\|^2 = \tilde{u}^T\tilde{u} = (\frac{f_t}{m})^2(\tilde{e}_z - e_z)^T(\tilde{e}_z - e_z). \tag{A.3}$$

The term $(\tilde{e}_z - e_z)^T(\tilde{e}_z - e_z)$ can be rewritten as

$$(\tilde{e}_z - e_z)^T(\tilde{e}_z - e_z) = 2(1 - \tilde{e}_z^T e_z). \tag{A.4}$$

Using that $\tilde{e}_z^T e_z = 1 - 2[(\tilde{q}_{v,1})^2 + (\tilde{q}_{v,2})^2]$, we get

$$\|\tilde{u}\|^2 = 4(\frac{f_t}{m})^2[(\tilde{q}_{v,1})^2 + (\tilde{q}_{v,2})^2]. \tag{A.5}$$

which completes the proof. □

APPENDIX B

THIRD ORDER FILTER OF REFERENCE TRAJECTORIES

The following section describes the filter used to obtain a smooth v_d from a possibly discontinuous $v_{d,raw}$. The transfer function of the utilized filter is as follows:

$$V_d(s) = \frac{\omega_n^2}{(Ts+1)(s^2+2\zeta\omega_n s+\omega_n^2)} V_{d,raw}(s) \tag{B.1}$$

where T, ω_n and ζ each denote the time constant, natural frequency and damping ratio of the filter, $V_{d,raw}(s)$ and $V(s)$ each denote the unfiltered v_d and filtered v_d in the Laplace domain. Similarly, the corresponding derivatives $\dot{v}_d$ and $\ddot{v}_d$ can be found with the following two transfer functions:

$$\dot{V}_d(s) = \frac{\omega_n^2 s}{(Ts+1)(s^2+2\zeta\omega_n s+\omega_n^2)} V_{d,raw}(s)$$
$$\ddot{V}_d(s) = \frac{\omega_n^2 s^2}{(Ts+1)(s^2+2\zeta\omega_n s+\omega_n^2)} V_{d,raw}(s).$$

Filters can be easily discretized to be used in digital micro-controllers. The three transfer functions shown above were converted in the form of a discrete 3×3 matrix before implementation.

APPENDIX C
EXPERIMENTAL EQUIPMENT

The implementation of the controller was done with a commercial quadrotor. The hardware only has minimal amount components to build a working quadrotor. The included components are as follows:

- Propellers $\times$ 4
- Rotors $\times$ 4
- Flight Controller
- External GPS/Compass Module
- Quadrotor Arm $\times$ 4
- Quadrotor Base and Cover
- ESC $\times$ 4
- LiPo Battery
- RC Reciever/Transmitter

In addition to the quadrotor components, a Serial Telemetry Radio receiver and transmitter were also used to establish connection between the quadrotor and a computer during flight. Table C.1 shows the individual components of the hardware. Figures C.1 to C.8 shows images of the individual components of the disassembled quadrotor. Finally, Figure C.9 shows the final assembly of the implemented quadrotor hardware.

Table C.1: Experimental Equipment

Type	**Description**
Flight Controller	Holybro Pix32 “Pixhawk”
GPS/Compass Module	Holybro Ublox Neo-M8N
Frame	DJI Flame Wheel F450
Motor	DJI 2312/960kV
ESC	DJI 420 Lite
Propeller	DJI 9450, Carbon Fiber
Battery	Flouren Li-Po RC Battery, 4S, 5500 mAh
RC Receiver/Transmitter	FlySky 2.4GHz 6 Channel Digital Transmitter and Receiver Radio System
Telemetry Receiver/Transmitter	Holybro 500mW FPV Transceiver Telemetry Radio Set V2, 433 MHz

Figure C.1: Flight Micro-controller

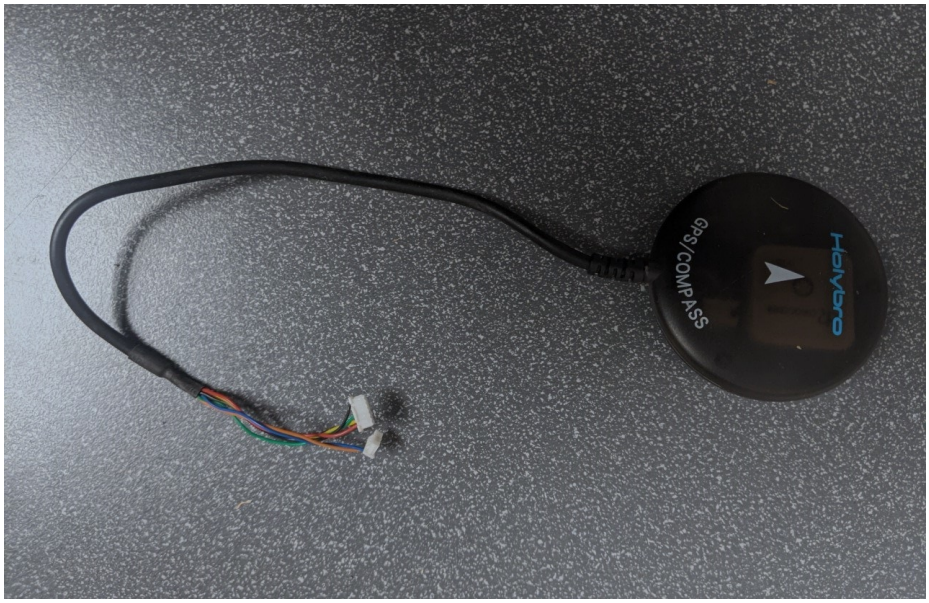


Figure C.2: External GPS Compass Module

Figure C.3: LiPo Battery and Frame Cover

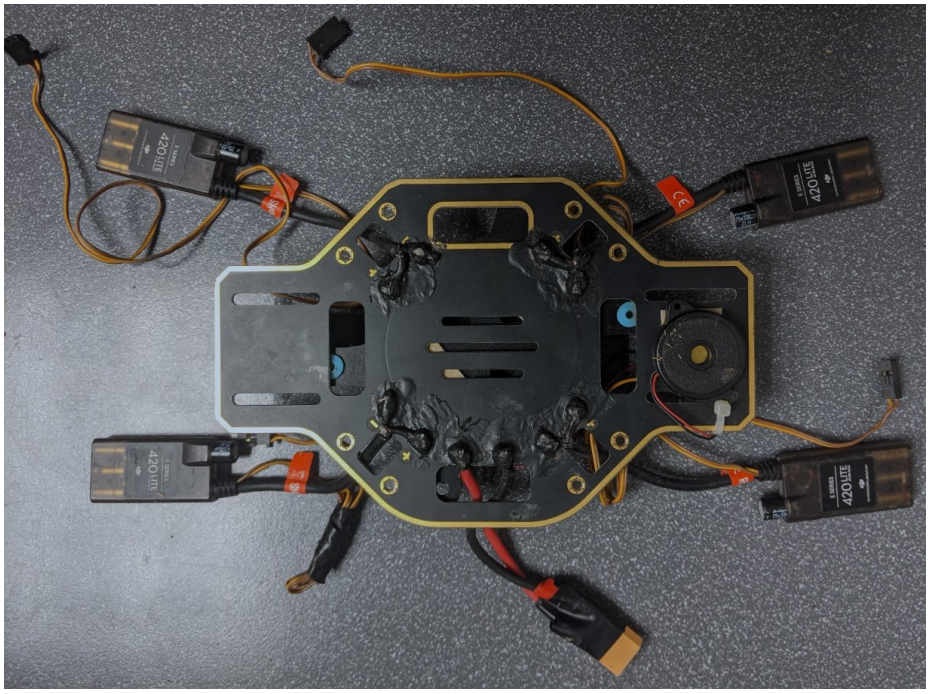


Figure C.4: Frame Base and ESC

Figure C.5: Frame Arm and Rotor

Figure C.6: Propellers

Figure C.7: RC Receiver and Transmitter

Figure C.8: Telemetry Receiver and Transmitter

Figure C.9: Euler Angles during Flipping Motion

REFERENCES

[1] D. Lee, H. Jin Kim, and S. Sastry, "Feedback linearization vs. adaptive sliding mode control for a quadrotor helicopter," *International Journal of Control, Automation and Systems*, vol. 7, no. 3, pp. 419–428, Jun. 2009.

[2] N. Cao and A. F. Lynch, "Inner–outer loop control for quadrotor UAVs with input and state constraints," *IEEE Transactions on Control Systems Technology*, vol. 24, no. 5, pp. 1797–1804, Oct. 2016.

[3] M. Schreier, "Modeling and adaptive control of a quadrotor," in *Proc. of the 2012 IEEE International Conference on Mechatronics and Automation*, Aug. 2012, pp. 383–390.

[4] D. E. Zlotnik and J. R. Forbes, "Rotation-matrix-based attitude control without angular velocity measurements," in *2014 American Control Conference*, 2014, pp. 4931–4936.

[5] F. Goodarzi and D. Lee, "Geometric adaptive tracking control of a quadrotor unmanned aerial vehicle on se(3) for agile maneuvers," *Journal of Dynamic Systems Measurement and Control*, vol. 137, Jun. 2015.

[6] J. T.-Y. Wen and K. Kreutz-Delgado, "The attitude control problem," *IEEE Transactions on Automatic Control*, vol. 36, no. 10, pp. 1148–1162, Oct. 1991.

[7] A. Tayebi and S. McGilvray, "Attitude stabilization of a VTOL quadrotor aircraft," *IEEE Transactions on Control Systems Technology*, vol. 14, no. 3, pp. 562–571, May 2006.

[8] E. Fresk and G. Nikolakopoulos, "Full quaternion based attitude control for a quadrotor," in *Proc. of the 2013 European Control Conference (ECC)*, Jul. 2013, pp. 3864–3869.

[9] H. Liu, X. Wang, and Y. Zhong, "Quaternion-based robust attitude control for uncertain robotic quadrotors," *IEEE Transactions on Industrial Informatics*, vol. 11, no. 2, pp. 406–415, Apr. 2015.

[10] J. Cariño Escobar, H. Abaunza González, and P. Castillo Garcia, "Quadrotor quaternion control," in *Proc. of the 2015 International Conference on Unmanned Aircraft Systems (ICUAS)*, Jun. 2015, pp. 825–831.

[11] M. Herrera, W. Chamorro, A. P. Gómez, and O. Camacho, "Sliding mode control: An approach to control a quadrotor," in *Proc. of the 2015 Asia-Pacific Conference on Computer Aided System Engineering*, Jul. 2015, pp. 314–319.

[12] K. Runcharoon and V. Srichatrapimuk, "Sliding mode control of quadrotor," in *2013 The International Conference on Technological Advances in Electrical, Electronics and Computer Engineering (TAEECE)*, 2013, pp. 552–557.

[13] B. Mu, K. Zhang, and Y. Shi, "Integral sliding mode flight controller design for a quadrotor and the application in a heterogeneous multi-agent system," *IEEE Transactions on Industrial Electronics*, vol. 64, no. 12, pp. 9389–9398, 2017.

[14] D. Xia, L. Cheng, and Y. Yao, "A robust inner and outer loop control method for trajectory tracking of a quadrotor," *Sensors*, vol. 17, p. 2147, Sep. 2017.

[15] T. Madani and A. Benallegue, "Backstepping control for a quadrotor helicopter," in *Proc. of the 2006 IEEE/RSJ International Conference on Intelligent Robots and Systems*, Oct. 2006, pp. 3255–3260.

[16] M. Huang, B. Xian, C. Diao, K. Yang, and Y. Feng, "Adaptive tracking control of underactuated quadrotor unmanned aerial vehicles via backstepping," in *Proceedings of the 2010 American Control Conference*, 2010, pp. 2076–2081.

[17] Y. Choi and H. Ahn, "Nonlinear control of quadrotor for point tracking: Actual implementation and experimental tests," *IEEE/ASME Transactions on Mechatronics*, vol. 20, no. 3, pp. 1179–1192, 2015.

[18] Z. Shulong, A. Honglei, Z. Daibing, and S. Lincheng, "A new feedback linearization LQR control for attitude of quadrotor," in *Proc. of the 2014 13th International Conference on Control Automation Robotics Vision (ICARCV)*, Dec. 2014, pp. 1593–1597.

[19] H. Voos, "Nonlinear control of a quadrotor micro-uav using feedback-linearization," in *2009 IEEE International Conference on Mechatronics*, 2009, pp. 1–6.

[20] G. V. Raffo, M. G. Ortega, and F. R. Rubio, "Backstepping/nonlinear h control for path tracking of a quadrotor unmanned aerial vehicle," in *2008 American Control Conference*, 2008, pp. 3356–3361.

[21] G. V. Raffo, M. G. Ortega, and F. R. Rubio, "An integral predictive/nonlinear h control structure for a quadrotor helicopter," *Automatica*, vol. 46, no. 1, pp. 29–39, 2010.

[22] A. Noormohammadi-Asl, O. Esrafilian, M. Ahangar Arzati, and H. D. Taghirad, "System identification and h-based control of quadrotor attitude," *Mechanical Systems and Signal Processing*, vol. 135, p. 106 358, 2020.

[23] B. T. Costic, D. M. Dawson, M. S. De Queiroz, and V. Kapila, "A quaternion-based adaptive attitude tracking controller without velocity measurements," in *Proc. of the*

39th IEEE Conference on Decision and Control (Cat. No.00CH37187), vol. 3, Dec. 2000, 2424–2429 vol.3.

[24] I. Palunko and R. Fierro, "Adaptive control of a quadrotor with dynamic changes in the center of gravity," *IFAC Proceedings Volumes*, vol. 44, no. 1, pp. 2626–2631, 2011, 18th IFAC World Congress.

[25] Y. Li and S. Song, "A survey of control algorithms for quadrotor unmanned helicopter," in *2012 IEEE Fifth International Conference on Advanced Computational Intelligence (ICACI)*, 2012, pp. 365–369.

[26] W. Lei, C. Li, and M. Z. Q. Chen, "Robust adaptive tracking control for quadrotors by combining pi and self-tuning regulator," *IEEE Transactions on Control Systems Technology*, vol. 27, no. 6, pp. 2663–2671, 2019.

[27] M. A. Lotufo, L. Colangelo, and C. Novara, "Control design for UAV quadrotors via embedded model control," *IEEE Transactions on Control Systems Technology*, pp. 1–16, 2019.

[28] T. Dierks and S. Jagannathan, "Output feedback control of a quadrotor uav using neural networks," *IEEE Transactions on Neural Networks*, vol. 21, no. 1, pp. 50–66, 2010.

[29] C. Nicol, C. J. B. Macnab, and A. Ramirez-Serrano, "Robust neural network control of a quadrotor helicopter," in *2008 Canadian Conference on Electrical and Computer Engineering*, 2008, pp. 001 233–001 238.

[30] J. Hwangbo, I. Sa, R. Siegwart, and M. Hutter, "Control of a quadrotor with reinforcement learning," *IEEE Robotics and Automation Letters*, vol. 2, no. 4, pp. 2096–2103, 2017.

[31] V. Murali, I. Spasojevic, W. Guerra, and S. Karaman, "Perception-aware trajectory generation for aggressive quadrotor flight using differential flatness," in *2019 American Control Conference (ACC)*, 2019, pp. 3936–3943.

[32] D. Mellinger, N. Michael, and V. Kumar, "Trajectory generation and control for precise aggressive maneuvers with quadrotors," *The International Journal of Robotics Research*, vol. 31, no. 5, pp. 664–674, 2012.

[33] H. Huang, G. M. Hoffmann, S. L. Waslander, and C. J. Tomlin, "Aerodynamics and control of autonomous quadrotor helicopters in aggressive maneuvering," in *2009 IEEE International Conference on Robotics and Automation*, 2009, pp. 3277–3282.

[34] J. Thomas, M. Pope, G. Loianno, E. W. Hawkes, M. A. Estrada, H. Jiang, M. R. Cutkosky, and V. Kumar, "Aggressive Flight With Quadrotors for Perching on In-

clined Surfaces," *Journal of Mechanisms and Robotics*, vol. 8, no. 5, May 2016, 051007.

[35] A. Castillo, R. Sanz, P. Garcia, W. Qiu, H. Wang, and C. Xu, "Disturbance observer-based quadrotor attitude tracking control for aggressive maneuvers," *Control Engineering Practice*, vol. 82, pp. 14–23, 2019.

[36] D. Cabecinhas, R. Cunha, and C. Silvestre, "A globally stabilizing path following controller for rotorcraft with wind disturbance rejection," *IEEE Transactions on Control Systems Technology*, vol. 23, no. 2, pp. 708–714, 2015.

[37] Z. Wang, X. Zhou, C. Xu, J. Chu, and F. Gao, "Alternating minimization based trajectory generation for quadrotor aggressive flight," *IEEE Robotics and Automation Letters*, vol. 5, no. 3, pp. 4836–4843, 2020.

[38] P. Yu, G. Chamitoff, and K. C. Wong, "Perching upside down with bi-directional thrust quadrotor," in *2020 International Conference on Unmanned Aircraft Systems (ICUAS)*, 2020, pp. 1697–1703.

[39] J. Kang, N. Sadegh, and C. Urschel, "Quaternion based nonlinear trajectory control of quadrotors with guaranteed stability," in *2020 American Control Conference (ACC)*, 2020, pp. 3834–3839.

[40] J. Diebel, "Representing attitude: Euler angles, unit quaternions, and rotation vectors," *Matrix*, vol. 58, Jan. 2006.

[41] S. Sarabandi and F. Thomas, "A survey on the computation of quaternions from rotation matrices," *Journal of Mechanisms and Robotics*, vol. 11, Oct. 2018.

[42] S. Abbott, "Understanding analysis," in, 2nd ed. Springer, 2016, ch. 4.

[43] K. Hassan K, "Nonlinear systems," in, 3rd ed. Upper Saddle River, NJ: Prentice-Hall, 2002, ch. 4.

[44] J. E. Slotine and W. Li, "Applied nonlinear control," in. Englewood Cliffs, NJ: Prentice-Hall, 1991, ch. 3.

[45] M. Ryll, H. H. Bülthoff, and P. R. Giordano, "Modeling and control of a quadrotor uav with tilting propellers," in *2012 IEEE International Conference on Robotics and Automation*, 2012, pp. 4606–4613.

[46] F. Riccardi, M. F. Haydar, S. Formentin, and M. Lovera, "Control of variable-pitch quadrotors," *IFAC Proceedings Volumes*, vol. 46, no. 19, pp. 206–211, 2013, 19th IFAC Symposium on Automatic Control in Aerospace.

[47] M. Fliess, J. Lévine, P. Martin, and P. Rouchon, "Flatness and defect of non-linear systems: Introductory theory and examples," *International Journal of Control*, vol. 61, pp. 13–27, Jun. 1995.

[48] D. Mellinger and V. Kumar, "Minimum snap trajectory generation and control for quadrotors," in *2011 IEEE International Conference on Robotics and Automation*, 2011, pp. 2520–2525.

[49] J. Yu, Z. Cai, and Y. Wang, "Minimum jerk trajectory generation of a quadrotor based on the differential flatness," in *Proceedings of 2014 IEEE Chinese Guidance, Navigation and Control Conference*, 2014, pp. 832–837.

[50] E. Tal and S. Karaman, "Accurate tracking of aggressive quadrotor trajectories using incremental nonlinear dynamic inversion and differential flatness," *IEEE Transactions on Control Systems Technology*, vol. 29, no. 3, pp. 1203–1218, 2021.

[51] B. Morrell, M. Rigter, G. Merewether, R. Reid, R. Thakker, T. Tzanetos, V. Rajur, and G. Chamitoff, "Differential flatness transformations for aggressive quadrotor flight," in *2018 IEEE International Conference on Robotics and Automation (ICRA)*, 2018, pp. 5204–5210.

[52] E. Stingu and F. Lewis, "Design and implementation of a structured flight controller for a 6dof quadrotor using quaternions," in *Proc. of the 2009 17th Mediterranean Conference on Control and Automation*, Jul. 2009, pp. 1233–1238.